Algorithmic Measures of Information for Tuples of Words and for Patterns in Multidimensional Shifts of Finite Type

Andrei Romashchenko

UNIVERSITÉ DE MONTPELLIER

Laboratoire d'Informatique, de Robotique et de Microélectronique de Montpellier
(UMR 5506 CNRS)

Mémoire d'Habilitation à Diriger les Recherches

Discipline : Informatique

par

Andrei ROMASHCHENKO

Algorithmic Measures of Information for Tuples of Words and for Patterns in Multidimensional Shifts of Finite Type.

Rapporteurs :	M. PETER GÁCS (BOSTON UNIVERSITY) M. JARKKO KARI (UNIVERSITY OF TURKU) M. YURY MAKARYCHEV (TOYOTA TECHNOLOGICAL INSTITUTE AT CHICAGO)
Date de soutenance :	19 November 2018
Composition du jury :	M. BRUNO DURAND (LIRMM – UNIVERSITÉ DE MONTPELLIER) M. EMERIC GIOAN (LIRMM – CNRS & UNIVERSITÉ DE MONTPELLIER) M. JEAN MAIRESSE (LIP6 – CNRS & UNIVERSITÉ PIERRE ET MARIE CURIE) M. WOLFGANG MERKLE (UNIVERSITÄT HEIDELBERG) M. NICOLAS OLLINGER (LIFO – UNIVERSITÉ D'ORLÉANS) M. NICOLAS SCHABANEL (LIP – CNRS & ENS DE LYON) M. ALEXANDRE ZVONKINE (LABRI – UNIVERSITÉ DE BORDEAUX)

THIS DOCUMENT IS A SHORT VERSION (WITH MINOR REVISIONS) OF THE MANUSCRIPT PRESENTED IN 2018 TO THE UNIVERSITY OF MONTPELLIER, FRANCE, IN ORDER TO OBTAIN THE *Habilitation à diriger des recherches*. THE AUTHOR THANKS THE REFEREES AND THE MEMBERS OF THE JURY FOR VALUABLE COMMENTS AND INTERESTING QUESTIONS.

Contents

Résumé du Mémoire

Le concept central de ce mémoire est la complexité de Kolmogorov et ses diverses applications. Le travail présenté s'articule autour de deux sujets principaux : d'un côté, nous étudions les mesures fondamentales de l'information, leurs propriétés universelles et leurs applications combinatoires ; de l'autre côté, nous étudions la notion de la densité de l'information dans le contexte de la dynamique symbolique. Bien que ces deux sujets soient liés l'un à l'autre, ils sont motivés par des questions très différentes et intéressent deux communautés scientifiques assez éloignées.

1. Le contexte autour de la théorie algorithmique de l'information

La notion de la complexité algorithmique a été introduite dans les années 1960. La formalisation mathématique de la notion de quantité d'information dans un objet individuel a été proposée en 1965 par Andrei Kolmogorov. Quelques années plus tôt, des idées similaires avaient été développées par Ray Solomonoff, qui était motivé par la notion de probabilité *a priori*. Des versions similaires de la complexité algorithmique ont été indépendamment proposées par Gregory Chaitin.

L'idée de la définition de Kolmogorov est très simple : la complexité $C(x)$ d'un objet constructif x est définie comme la longueur du programme le plus court qui produit cet objet. De la même manière, la complexité conditionnelle $C(x \mid y)$ d'un objet x relativement à un autre objet y est définie comme la longueur du programme le plus court qui transforme y en x. L'ambiguïté causée par le choix du langage de programmation est résolue par un théorème d'invariance, qui garantit l'existence d'un langage de programmation optimal : un langage optimal assigne à chaque objet une valeur de la complexité qui est minimale à une constante additive près.

Bien que la complexité de Kolmogorov soit normalement définie pour les mots binaires, cette définition peut être facilement transposée à d'autres types d'objets finis : les mots sur tout alphabet, les suites finies de mots, les matrices, les ensembles finis, les graphes finis, etc.

L'article original de Kolmogorov était intitulé « Trois approches à la définition du concept de quantité d'information ». Ces trois approches ont été respectivement algorithmiques, probabilistes et combinatoires. L'approche algorithmique repose sur la complexité algorithmique (complexité de Kolmogorov) mentionnée ci-dessus. L'approche probabiliste est attribuée à Claude Shannon, qui a défini en 1948 la notion d'entropie. L'entropie de Shannon mesure la quantité d'information dans une variable aléatoire. Cette approche est à l'heure actuelle la plus populaire : elle

est très efficace à la fois en théorie et dans les applications pratiques ; plusieurs chercheurs et ingénieurs spécialisés dans la théorie de l'information travaillent dans le cadre de la méthode suggérée par Shannon. L'approche combinatoire à la notion d'information (attribuée à Ralph Hartley) mesure l'incertitude d'un élément inconnu dans un ensemble fini. Plus précisément, cette mesure est définie comme logarithme de la cardinalité de l'espace de messages possibles. Malgré sa simplicité apparente, cette approche est loin d'être triviale. Nous verrons dans ce mémoire que l'idée de l'information « combinatoire » de Hartley amène très naturellement notre recherche à des problèmes intéressants et sophistiqués.

Kolmogorov a mis en évidence des analogies et des liens étroits entre ces trois « incarnations » de la théorie de l'information. Même aujourd'hui, 50 ans après la publication originale de Kolmogorov, les interactions entre les approches de Hartley, Shannon et Kolmogorov restent un fil conducteur pour les chercheurs travaillant sur la théorie de l'information.

Nous pouvons étudier le parallélisme entre les théories de Shannon et de Kolmogorov sous différents angles. Dans certains cas, nous pouvons établir une équivalence précise et formelle entre les résultats probabilistes et les résultats probabilistes algorithmiques. Alors, certaines propriétés de la théorie algorithmique de l'information peuvent être formellement traduites en propriétés équivalentes de la théorie de l'information de Shannon et réciproquement. Ensuite, les méthodes d'une théorie peuvent être réutilisées directement pour déduire de nouveaux résultats dans l'autre. Dans d'autres cas, il n'y a pas d'équivalence formelle entre les théorèmes sur l'entropie de Shannon et la complexité de Kolmogorov, mais le parallélisme entre ces deux théories nous donne des heuristiques utiles. Alors, étant conscient des théorèmes d'une incarnation de la théorie de l'information, nous pouvons deviner quels résultats peuvent également être vrais dans l'autre théorie, malgré l'absence d'une équivalence formelle et explicite. Dans certains cas, le parallélisme entre les différentes versions de la théorie de l'information échoue totalement, et nous observons une discordance entre les propriétés des mesures de l'information dans les contextes algorithmique et probabiliste. Cependant, même ce genre d'échec est fructueux : en explorant les raisons de cette divergence, nous révélons des propriétés fines de la complexité de Kolmogorov et de l'entropie de Shannon.

2. Les mesures fondamentales dans la théorie algorithmique de l'information

Une partie importante du mémoire est consacrée aux mesures basiques d'information qui sont utilisées dans les versions algorithmique et probabiliste de la théorie de l'information. Pour chaque ensemble de mots binaires, nous nous sommes donné plusieurs quantités standards d'information : il s'agit des complexités de Kolmogorov de chaque mot, des complexités de chaque paire de mots, des triplets, et ainsi de suite, ainsi que de nombreuses instances de la complexité de Kolmogorov conditionnelle, aussi bien que les valeurs de l'information mutuelles, etc. Également, étant donné un ensemble de variables aléatoires (distribuées conjointement), nous nous sommes donné les valeurs de l'entropie de Shannon de chaque distribution

marginale, de nombreuses instances de l'entropie conditionnelle, des instances de l'information mutuelle, et ainsi de suite. Le problème fondamental est donc de comprendre les liens entre différentes quantités d'information.

Plusieurs relations fondamentales entre les quantités d'information sont bien connues depuis les publications originales de Shannon (1948) et Kolmogorov (1965). Par exemple, nous savons que l'entropie d'une paire de variables aléatoires ne peut jamais être plus grande que la somme des entropies de ses composantes. De même, la complexité de Kolmogorov d'une paire de mots n'est pas plus grande (à un terme négligeable près) que la somme des complexités des deux mots,

$$C(x,y) \lesssim C(x) + C(y).$$

Cette propriété (appelée *subadditivité*) ainsi que les propriétés standards de *monotonicité* et *submodularité* possèdent des interprétations intuitives claires, ce qui est apparemment important pour de nombreuses applications de la théorie de l'information. Pendant plusieurs années ces trois inégalités fondamentales se sont manifestées dans de nombreuses applications de l'entropie de Shannon, en rattachant l'intuition des ingénieurs et les outils formels des mathématiciens. En 1998 (50 ans après la publication originale de Shannon), cette unicité de l'intuition pratique et de la théorie mathématique a été mise en cause : Z. Zhang et R. W. Yeung ont trouvé le premier exemple d'une inégalité linéaire pour l'entropie de type non-Shannon. Cette inégalité est vraie pour chaque quadruple de variables aléatoires (distribuées conjointement), bien qu'elle ne puisse pas être représentée comme une combinaison des inégalités classiques (qui expriment les propriétés de subadditivité, de monotonie et de submodularité de l'entropie de Shannon).

Inégalités d'information. Au cours des vingt dernières années, de nombreux autres exemples d'inégalités d'information non classiques (pour l'entropie de Shannon) ont été trouvés. Bien que la caractérisation complète des inégalités universelles pour l'entropie de Shannon reste insaisissable, nous savons maintenant qu'exactement les mêmes classes d'inégalités linéaires sont justes pour l'entropie de Shannon et pour la complexité de Kolmogorov (le résultat prouvé dans la thèse de doctorat de l'auteur de ce mémoire). L'équivalence établie suggère que ces inégalités (qui sont communes pour les versions probabiliste et algorithmique de la théorie de l'information) expriment des propriétés de la notion de l'information qui sont vraiment fondamentales. Nous croyons que la poursuite des recherches sur les inégalités informationnelles reste une direction intéressante et prometteuse.

Dans une publication jointe avec Konstantin et Yury Makarychev et Nikolai Vereshchagin, nous avons proposé une nouvelle méthode pour dériver les inégalités d'information. Cette technique utilise le lien entre les inégalités d'information et le concept de l'extraction de l'information mutuelle (qui remonte aux publications de R. Ahlswede, P. Gács, et J. Körner dans les années 1970). Ce type d'argument a permis de retrouver une interprétation intuitive de l'inégalité non classique de Zhang et Yeung et de découvrir une série infinie de nouvelles inégalités d'information.

Depuis la fin des années 1990, les chercheurs ont découvert quelques exemples des inégalités dites *inégalités conditionnelles d'information* — des inégalités

linéaires pour l'entropie de Shannon qui sont valides pour les distributions vérifiantes des conditions linéaires pour les valeurs d'entropies. Dans une série de travaux communs avec Tarik Kaced nous avons montré que certaines de ces inégalités sont, pour ainsi dire, *substantiellement conditionnelles*. Cela signifie qu'elles ne peuvent pas être obtenues comme une projection d'une inégalité d'information conventionnelle (inconditionnelle) quelconque, connue ou encore inconnue. Nous avons établi la connexion entre les inégalités conditionnelles et la géométrie du cône des inégalités inconditionnelles pour l'entropie de Shannon. Plus précisément, nous avons montré qu'une seule inégalité conditionnelle pour un ensemble de variables aléatoires correspond à une infinité d'inégalités linéaires conventionnelles. L'existence d'une inégalité conditionnelle de ce type explique pourquoi le cône d'inégalités inconditionnelles pour n variables aléatoires (avec $n \geq 4$) n'est pas polyédrique (un résultat prouvé plus tôt par F. Matúš). Nous avons étudié les contreparties des inégalités conditionnelles d'information dans le cadre de la complexité de Kolmogorov. Nous avons trouvé que certaines de ces inégalités possèdent des homologues valides pour la complexité de Kolmogorov, tandis que d'autres ne les possèdent pas. Apparemment, cette divergence entre les propriétés de l'entropie de Shannon et de la complexité de Kolmogorov mérite une grande attention. En explorant ce type d'inégalités conditionnelles « exceptionnelles » (qui sont valables dans le cadre de Shannon mais pas dans celui de Kolmogorov), nous avons obtenu des applications intéressantes de la théorie de l'information à l'analyse combinatoire et à la complexité de la communication, comme décrit ci-dessous.

Matérialisation de l'information mutuelle et de la complexité conditionnelle.
Pour chaque paire de mots x, y nous nous sommes donné les complexités conditionnelles $C(x \mid y)$ et $C(y \mid x)$ (qui sont par définition les longueurs des programmes les plus courts qui transforment x en y et y en x), et la valeur de l'information mutuelle $I(x : y)$, qui est définie sous une forme symétrique telle que

$$I(x : y) := C(x) + C(y) - C(x, y)$$

(le « chevauchement » des informations contenues dans x et dans y). Dans ce contexte, nous pouvons poser la question sur l'interprétation plus explicite et opérationnelle de ces quantités d'information. Dans le cadre de la théorie de Shannon, le théorème de Slepian–Wolf donne une caractérisation opérationnelle de l'entropie conditionnelle, tandis que Gács et Körner ont montré que l'information mutuelle en général ne peut pas être matérialisée. Dans le cadre de la théorie algorithmique, cette question est devenue populaire dans les années 1990. Dans ce qui suit, nous discutons séparément le problème de la matérialisation de la complexité conditionnelle et le problème de la matérialisation de l'information mutuelle.

À la fin des années 1990 An. Muchnik a proposé une interprétation intéressante de la valeur de $C(x \mid y)$. Par définition, cette quantité est comprise comme la longueur du programme le plus court qui transforme le mot y en mot x. Selon Muchnik, cette quantité peut aussi être interprétée comme une « empreinte digitale numérique » optimale (ou « valeur de hachage »), qui (i) peut être extraite directement du mot x, et (ii) qui donne assez d'information pour reconstruire x si y est

donné. En d'autres termes, la quantité de $C(x \mid y)$ peut être « extraite » directement de x, quasiment sans autre information sur y.

Dans un travail commun avec Daniil Musatov et Alexander Shen, nous avons présenté deux nouvelles preuves de ce théorème. L'une de nos constructions est basée sur un algorithme qui cherche « *online matching* » pour les graphes bipartis. La seconde, basée sur des *graphes extracteurs*, permet de démontrer une version du théorème de Muchnik pour la complexité de Kolmogorov avec une limite d'espace (mémoire) accessible au calcul. Nous avons prouvé aussi une autre version du théorème de Muchnik avec des ressources limitées — pour la complexité de Kolmogorov basée sur les protocoles d'Arthur–Merlin. Dans les travaux ultérieurs de M. Zimand, une autre version du théorème de Muchnik a été établie : Zimand a montré que l'empreinte digitale numérique de x peut être calculée très efficacement, en temps polynomial. (Pourtant la reconstruction de x à partir de y et l'empreinte digitale numérique reste coûteuse et prend beaucoup de temps.)

Les techniques utilisées dans les preuves des différentes variantes du théorème de Muchnik impliquent certaines versions de graphes *pseudo-aléatoires* (extracteurs, expandeurs). La technique des empreintes digitales pseudo-aléatoires développée dans le contexte du théorème de Muchnik a trouvé une application dans un domaine très différent : nous l'avons utilisée pour construire une structure de données (un schéma de *bit-probe* avec un traitement randomisé des requêtes) avec des paramètres presque optimaux. Cette structure stocke un ensemble fini S sous une forme compressée, de sorte que les requêtes d'appartenance « $x \in S$? » sont traitées pour chaque x très rapidement, en récupérant seulement un bit de la base de données.

Le problème de la « matérialisation » de l'information mutuelle des mots est plus délicat que la matérialisation de la complexité conditionnelle. L'intuition naïve suggère que l'information mutuelle entre x et y représente la quantité de données partagée par deux mots. Les chercheurs ont donc exploré la mesure dans laquelle l'information mutuelle peut être extraite de x et y, d'une manière ou d'une autre. P. Gács et J. Körner ont introduit dans ce contexte le terme *information commune*. Bien que la définition précise soit assez technique, l'intuition derrière l'information commune est simple. Dans les grandes lignes, z est une information commune extraite de x et y, si z peut être « facilement » calculé à partir de x et à partir de y, i.e. $C(z \mid x) \approx 0$ et $C(z \mid y) \approx 0$.

Il est connu que l'information *commune* est limitée par l'information *mutuelle*. Pourtant, Gács et Körner (1973) ont montré qu'il existe des mots x et y tels que l'information commune entre eux est beaucoup plus petite que l'information mutuelle. De plus, l'écart entre l'information mutuelle et l'information commune ne dépend pas uniquement des mesures de complexité de x et y. En fait, il existe des paires (x_1, y_1) et (x_2, y_2) qui ont le même profil de complexité, mais pour (x_1, y_1) l'information commune et l'information mutuelle sont égales, alors que pour (x_2, y_2) elles sont radicalement différentes. Des résultats négatifs similaires (l'information commune peut être de beaucoup inférieure à l'information mutuelle, même avec un seuil plutôt lâche pour les valeurs de « petites » complexités $C(z \mid x)$ et $C(z \mid y)$) ont été établies dans les années 1990 par un groupe de chercheurs à l'université de Moscou (Muchnik, Shen, Vereshchagin et d'autres ; quelques résultats de ce type

ont été présentés dans la thèse de doctorat de l'auteur de ce mémoire).

Nous pouvons reformuler le problème de l'extraction de l'information mutuelle pour un triplet de mots. Nous disons que l'information mutuelle partagées par des mots x, y, z peut être complètement matérialisée, s'il existe un mot w tel que x, y et z sont indépendants étant donné w, et w est simple conditionnel relativement à chacun des mots x, y, z. Nous avons montré que la question de l'extraction de l'information mutuelle pour un triplet de mots a une réponse positive surprenante : l'information mutuelle partagée par des mots x, y, et z peut être complètement matérialisée, si et seulement si, les valeurs de l'information mutuelle

$$I(x : y \mid z),\ I(x : z \mid y), \text{ et } I(y : z \mid x)$$

sont négligeables. La preuve de ce résultat dépend fortement des nouvelles inégalités d'information non-classiques mentionnées ci-dessus.

Applications combinatoires. L'entropie de Shannon est un outil qui s'applique bien dans les preuves combinatoires, y compris dans des cas où l'idée de la quantité de l'information ne se manifeste pas immédiatement. Un exemple classique d'un tel argument est le lemme combinatoire de Shearer, qui est, en substance, une traduction d'une inégalité pour l'entropie de Shannon dans le langage de l'information « combinatoire » de Hartley (autrement dit, en termes de tailles d'ensembles finis). Cependant, les applications combinatoires de la théorie de l'information ressemblent généralement à une solution *ad hoc*. Dans le même temps, l'idée du parallélisme entre trois approches à la théorie de l'information proposée par Kolmogorov suggère qu'une méthode plus générale doit exister. En particulier, nous pouvons supposer que toute inégalité d'information (valide à la fois pour l'entropie de Shannon et pour la complexité de Kolmogorov) pourrait être traduite en forme combinatoire équivalente. Dans un travail commun avec A. Shen et N. Vereshchagin nous avons essayé d'établir formellement cette règle de traduction. Nous avons démontré que toute inégalité linéaire valide pour la complexité de Kolmogorov ou l'entropie de Shannon peut être reformulée comme une propriété combinatoire. Les inégalités combinatoires de Shearer peuvent être considérées comme un cas particulier de cette traduction générale. Plus tard, une autre version d'une interprétation combinatoire des inégalités d'information a été suggérée par A. Shen et N. Vereshchagin dans un travail commun avec I. Newman et G. Tárdos.

En ce qui concerne les inégalités conditionnelles d'informations, aujourd'hui, nous n'avons pas de schéma général qui traduise chaque inégalité en une proposition combinatoire équivalente. Cependant, nous avons trouvé des applications combinatoires pour certaines (très particulières) inégalités contraintes d'informations. Les propriétés combinatoires que nous avons obtenues peuvent être formulées naturellement en termes de coloration des arêtes des graphes (un travail commun avec T. Kaced et N. Vereshchagin). De ce point de vue, certaines inégalités contraintes d'information sont comprises comme des inégalités pour les couvertures de graphe par bicliques (en résumé, nous pouvons estimer la quantité de bicliques nécessaires pour couvrir toute arête d'un graphe donné). Cette technique s'applique dans les preuves de résultats négatifs sur la complexité de la communication, comme décrit ci-dessous.

Une caractérisation opérationnelle de l'information mutuelle. Dans un travail commun avec Marius Zimand, nous avons réexaminé la question de la matérialisation de l'information mutuelle dans la théorie de l'information algorithmique et proposé une sorte de « caractérisation opérationnelle » de cette valeur. Nous montrons que l'information mutuelle d'une paire de mots x et y est égale à la taille de la plus longue clé secrète commune que deux parties (l'une ayant x et l'autre ayant y) peuvent établir via un protocole probabiliste de communication, avec interaction sur un canal ouvert. Ce résultat est généralisé au cas de n parties, $n > 2$. Dans ce travail, nous avons employé plusieurs résultats de la théorie algorithmique de l'information (qui sont mentionnées ci-dessus) : le théorème de Muchnik sur la complexité conditionnelle permet de construire un protocole de communication, qui atteint la taille optimale de la clé secrète commune ; les inégalités conditionnelles d'information nous aident à prouver les résultats négatifs (la taille de la clé commune ne peut pas être plus grande que l'information mutuelle entre les mots d'entrées) ; et enfin, les résultats négatifs sur l'extraction de l'information commune sont utilisés pour estimer la complexité de la communication de ce problème (si le nombre des bits transmis tombe en dessous du seuil établi, alors seulement une clé secrète très petite peut être obtenue).

3. Shifts de type fini : les règles locales et la densité de l'information

Le second chapitre du manuscrit est consacré aux shifts multidimensionnels, surtout à leurs propriétés algorithmiques. Les *shifts* sont les ensembles de configurations invariants par la translation et topologiquement clos dans l'espace de Cantor (plus précisément, les ensembles de configurations dans $\mathbb{Z}^d$ sur un alphabet fini, invariants par la translation et topologiquement clos). Chaque shift est uniquement déterminé par un ensemble de *motifs interdits* (de sorte qu'une configuration appartienne au shift, si et seulement si, elle n'implique aucun motif interdit). Les *shifts de type fini* (*shifts of finite type*, SFT) sont les shifts définis par un nombre fini de motifs interdits.

C'est ici qu'il faut mentionner un type spécial des SFTs, les *pavages de Wang* (Wang tilings), qui sont définis en termes de tuiles. En dimension deux, les *tuiles de Wang* sont normalement représentées comme un ensemble de carrés colorés sur chacun de leurs côtés. Les carrés peuvent être agencés de telle façon que les couleurs des côtés correspondants soient les mêmes. (Il existe, bien évidemment, une généralisation des tuiles de Wang à l'espace à dimension $d > 2$.) Cette définition ressemble à un jeu d'enfant (une version simple d'un puzzle). Malgré la simplicité de la définition, ce modèle est très général : chaque shift de type fini est isomorphe à un shift qui correspond à un jeu de tuiles de Wang.

Les shifts de type fini donnent un cadre pour étudier l'apparition de phénomènes non-triviaux globaux à partir de règles locales et très simples. Les shifts de type fini jouent un rôle important dans plusieurs domaines, tels que la logique mathématique, la théorie des langages, la complexité de calcul, la dynamique symbolique et même la physique mathématique (où les shifts de type fini sont utilisés comme un modèle simpliste pour des structures cristallines et quasi-cristallines).

Plusieurs résultats profonds concernant les shifts de type fini dépendent fortement de l'incorporation d'un calcul dans la structure géométrique d'un shift. Nous proposons une nouvelle technique d'intégration d'une machine de Turing dans un shift de type fini. Cette technique est connue comme les *pavages auto-simulants* ou les *pavages de point-fixe*. Notre méthode réunit l'idée de l'auto-similitude géométrique et de l'idée d'un programme auto-référentiel.

Notre technique de programmation auto-référentielle est similaire à la preuve classique du théorème de récursion de Kleene. Il existe deux autres exemples dans la littérature qui sont encore plus proches de notre construction : la construction des automates autoréplicatifs de J. von Neumann et les automates cellulaires autocorrecteurs de P. Gács. Il s'est avéré que la méthode de la programmation auto-référentielle se marie parfaitement à l'idée géométrique d'une configuration auto-similaire, qui reproduit des structures similaires à différentes échelles. En employant cette technique, nous proposons de nouvelles preuves de certains théorèmes classiques sur la dynamique symbolique, et nous obtenons ensuite quelques nouveaux résultats. Ci-dessous, nous discutons brièvement les directions principales de cette recherche.

De règles locales simples vers un comportement global complexe. Pour tout $d \geq 2$ il existe un shift de type fini non-vide en dimension d, où toute configuration est apériodiques (R. Berger, 1966). Le fait de l'existence d'un SFT apériodique est déjà non-trivial. De façon encore plus importante, des shifts de type fini apériodiques sont utilisés comme la base et le cadre pour plusieurs constructions avec diverses propriétés non-triviales.

La construction proposée par Berger en 1966 est assez complexe. Dans la suite, plusieurs preuves plus simples de ce résultat ont été suggérées. Toutes les constructions connues d'un SFT apériodique combinent dans des proportions différentes des idées géométriques et algorithmiques. Notre construction (basée sur l'idée de l'auto-simulation) est en quelque sorte extrémale : elle est presque purement algorithmique, avec très peu d'éléments géométriques. Cette approche ne permet pas de minimiser la taille de l'alphabet ou le nombre de contraintes locales d'un SFT construit. Cependant, elle est assez flexible et admet diverses généralisations et extensions, ce qui est crucial pour les applications.

Nous commençons par de nouvelles preuves de certains résultats déjà connus. Une des premières applications de la nouvelle technique est une preuve simplifiée du résultat classique par W. Hanf et D. Myers (1974) : pour chaque $d \geq 2$ il existe un shift de type fini non-vide où toute configuration est non-calculable. La même technique est employée ensuite pour construire un shift de type fini avec une densité d'information maximale. Plus précisément, pour chaque $d \geq 2$, nous construisons un shift de type fini de dimension d, où pour chaque configuration, chaque motif de taille $n \times n \times \cdots \times n$ a la complexité de Kolmogorov $\Omega(n^{d-1})$. (Un shift de type fini avec une densité d'information élevée a été construit à l'origine par B. Durand, L. Levin, et A. Shen (2001) par une technique basée sur les pavages de Robinson.)

Une extension du résultat mentionné ci-dessus (un shift de type fini avec une haute complexité de Kolmogorov) nous a conduit à une meilleure compréhension

de la sous-dynamique projective des shifts de type fini multidimensionnels. Nous avons montré que tout shift effectif sur $\mathbb{Z}^d$ peut être représenté comme une projection de la sous-dynamique dans un shift de type fini sur $\mathbb{Z}^{d+1}$. Cette observation répond à une question soulevée par M. Hochman (2009). Ce dernier résultat a été prouvé indépendamment par N. Aubrun et M. Sablik (2013) avec une autre technique.

SFTs quasi-périodiques minimaux. A. Ballier et N. Ollinger (2009) ont montré qu'il existe un shift de type fini où chaque configuration est à la fois apériodique et quasi-périodique. Nous avons suggéré une nouvelle preuve de ce résultat, en utilisant des pavages auto-simulants. Il est à noter que dans cet exemple particulier nous avons employé une technique intrinsèquement algorithmique (la programmation récursive, des programmes auto-référentiels) pour prouver un théorème sans aucune notion algorithmiques dans son assertion.

Nous avons également prouvé qu'il existe un shift de type fini où chaque configuration est à la fois non-calculable et quasi-périodique. De plus, nous avons obtenu une caractérisation des classes de degrés de Turing qui peuvent être représentées par les configurations d'un shift de type fini quasi-périodique. À notre connaissance, les seules preuves connues de ces résultats utilisent la technique des pavages auto-simulants.

SFTs robustes (tolérants aux fautes locales). Nous montrons que plusieurs résultats mentionnés ci-dessus (les shifts de type fini apériodiques et fortement apériodiques, les shifts de type fini avec des motifs à haute complexité de Kolmogorov, et ainsi de suite) restent valides dans un contexte plus général — pour les pavages avec des « fautes », c'est-à-dire pour les configurations où les contraintes locales peuvent être perturbées par un ensemble clairsemé d' « erreurs » choisies au hasard. La technique des pavages auto-simulants est, jusqu'à présent, la seule méthode connue pour construire ce type de shifts de type fini robustes. Dans certains de ces résultats, nous observons à nouveau qu'une technique fortement algorithmique peut être employée dans les démonstrations de théorèmes mathématiques sans objets algorithmiques dans leurs assertions.

Remarques de conclusion. La technique des pavages auto-simulants est apparue dans une série d'articles communs avec Bruno Durand et Alexander Shen. Notre travail a été largement inspiré par la construction d'automates cellulaires fiables suggérée par P. Gács. Au cours des dernières années, la technique des pavages auto-simulants a été redéveloppée par L. B. Westrick et C. Zinoviadis.

Nos contributions : les techniques proposées et l'interrelation entre les domaines apparentés

Dans le premier chapitre du mémoire nous étudions les visages divers des mesures quantitatives de l'information. Nous examinons les définitions d'un objet aléatoire et pseudo-aléatoire sous différentes perspectives. Notre étude est motivée par des

parallèles entre les approches au concept de l'information proposées et développées dans les différentes communautés. Nous employons des idées similaires dans des contextes plutôt hétérogènes, avec des techniques probabilistes, algorithmiques et combinatoires. D'un point de vue technique, nous proposons quelques méthodes nouvelles (par exemple, la technique du « clonage » pour la complexité de Kolmogorov, les inégalités conditionnelles d'information) ; mais notre contribution principale consiste en l'adaptation des techniques développées dans la théorie de l'information (inégalités d'information, extraction de l'information commune, graphes pseudo-aléatoires, etc.) dans des situations variées. Dans le texte du mémoire, nous discutons les motivations et les idées principales derrière les preuves de théorèmes. Normalement, les preuves détaillées sont reportées aux publications originales (certaines d'entre elles sont jointes en annexe). Cependant, nous présentons dans le texte du mémoire certaines esquisses des preuves techniques, dans les cas où nous croyons qu'une perte (légère) de généralité permet de simplifier considérablement l'argument concerné.

L'organisation du second chapitre du mémoire est différente. Ce chapitre s'articule autour d'une technique particulière — la construction des pavages auto-simulants. Nous étudions en détail des aspects divers de cette technique, depuis la construction générique vers plusieurs généralisations et renforcements. Nous parlons d'applications dans lesquelles cette méthode est efficace et discutons ses limites. Tandis que dans le premier chapitre nous réunissons des techniques variées avec une intuition commune, dans le second chapitre nous faisons le contraire — des techniques similaires s'appliquent à des situations hétérogènes.

Dans cette ligne de recherche, il est difficile de bien séparer la technique de base des suppléments spécifiques pour chaque application particulière. C'est pourquoi dans tout article original, nous sommes obligés de réexpliquer la technique de base de pavages auto-simulants. Dans ce chapitre, nous préférons ne pas renvoyer le lecteur vers des publications originales où les parties techniques se chevauchent fortement. Au lieu de cela, nous présentons les preuves complètes de nos résultats principaux.

Les sujets du chapitre 1 et du chapitre 2 se chevauchent peu, et formellement parlant, ces chapitres peuvent être lus indépendamment l'un de l'autre. Pourtant, ils sont réunis par un concept fondamental qui joue un rôle important dans l'un et l'autre chapitre : c'est la complexité de Kolmogorov, qui donne la mesure de quantité de l'information dans un objet fini pour le premier chapitre et la mesure de densité d'information pour les configurations infinies pour le second. En conclusion du mémoire nous proposons un projet de travail ultérieur. Dans ce projet nous traçons quelques lignes de recherche (basées sur la complexité de Kolmogorov avec des ressources bornées) qui pourraient éventuellement réunir ces deux domaines plus étroitement.

Introduction and research summary

The main theme of this manuscript is Kolmogorov complexity and its various applications. Our research is mostly related to the following two areas: the fundamental information measures and their combinatorial applications on the one hand, and the notion of information density in symbolic dynamics on the other. Although these research areas are closely related, they are driven by different motivating factors and are attributed to two different scientific communities.

1. The context around algorithmic information theory

The notion of algorithmic complexity was introduced in the theory of computability in the 1960s. The formal mathematical definition of the amount of information in a constructive object was suggested in 1965, in the seminal paper by Andrei Kolmogorov. A few years earlier, similar ideas appeared in papers by Ray Solomonoff (who was studying the notion of the universal *a priori* probability). Also, similar versions of algorithmic complexity were proposed by Gregory Chaitin.

The intuition behind Kolmogorov's definition is quite transparent: the complexity of a constructive object is the length of the shortest program that generates this object. Similarly, the complexity of one object given another one is the length of the shortest program that transforms the latter into the former. The ambiguity caused by the choice of the programming language is resolved by the invariance theorem, which claims that there exists an asymptotically optimal programming language that assigns to all objects an asymptotically minimal complexity. The Kolmogorov complexity is usually defined for binary strings, but the standard definition can be easily transposed to other types of finite objects: words over any finite alphabet, tuples of words, matrices, finite sets, finite graphs, etc.

The original paper by Kolmogorov was entitled "Three Approaches to the Quantitative Definition of Information." These approaches were the algorithmic, the probabilistic, and the combinatorial ones. The probabilistic approach goes back to Claude Shannon who defined in 1948 the notion of entropy, which is a measure of information quantity in a random variable. This approach has proven to be highly fruitful both in theory and in practical applications, and the vast community of information theorists work by now in the framework suggested by Shannon. The combinatorial approach to the notion of information (traced back to Ralph Hartley) measures the "uncertainty" of an element in a finite set; technically this measure is defined as the logarithm of its cardinality. Despite its apparent simplicity, the combinatorial approach is far from trivial; Hartley's definition of information in finite sets naturally lead to interesting and sophisticated mathematical problems.

Kolmogorov highlighted the parallels and close connections between these three incarnations of information theory. Even 50 years after the original paper by Kolmogorov, the interplay between Hartley's, Shannon's, and Kolmogorov's approaches remains a major clue for information theorists. The parallelism between Shannon's and Kolmogorov's theories can be viewed from different perspectives. In some cases we can establish a formal equivalence between probabilistic and algorithmic results, i.e., some types of properties of algorithmic information theory can be for-

mally translated to equivalent properties of Shannon's information theory, and visa-versa. Then, the technique from one theory can be reemployed directly to deduce new results in the other one. In other cases, there are no formal equivalences between the frameworks of Shannon's entropy and Kolmogorov complexity, but the parallelism provides helpful heuristics: knowing a result from one incarnation of the information theory we can guess the homologous statement in the other, even if there is no explicit formal equivalence between the statements. In some cases, the parallelism between different versions of information theory fails and we observe a discrepancy between the properties of information measures in the algorithmic and the probabilistic frameworks. But even this kind of failure is fruitful: while exploring the reasons for that discrepancy we reveal deeper properties of Kolmogorov complexity and Shannon's entropy.

2. The fundamental measures in algorithmic information theory

A major part of our research is devoted to the information measures in algorithmic and probabilistic versions of information theory. Given a tuple of strings, we can deal with many standard information quantities: with the Kolmogorov complexities of each of these strings, with the complexities of each pair, triple, and so on, with many instances of conditional Kolmogorov complexities, with the values of pairwise mutual information, and so on. Similarly, given a jointly distributed tuple of random variables, we have the values of Shannon's entropy of each marginal distribution, many instances of the conditional entropy, instances of the pairwise mutual information, and so on. The fundamental problem is to understand the connection between different information quantities.

Several basic relations between information quantities are well known since the original papers by Shannon and Kolmogorov. For example, it is known that the entropy of a pair of random variables cannot be larger than the sum of entropies of two its components. Similarly, the Kolmogorov complexity of a pair of strings is not greater (up to a negligible additive term) than the sum of complexities of both strings. This property (called *subadditivity*) as well as the standard properties of *monotonicity* and *submodularity* have clear intuitive meanings, which is important for numerous applications of information theory. In 1998 (50 years after Shannon's original publication), Z. Zhang and R.W. Yeung came up with the first example of a non-Shannon-type linear inequality for the entropies of a tuple of random variables. This inequality holds true for every quadruple of jointly distributed random variables, though it cannot be represented as a convex combination of the basic inequalities (expressing the properties of subadditivity, monotonicity, and submodularity of Shannon's entropy).

Information inequalities. In the last 20 years, many other examples of non-classical information inequalities for Shannon's entropy have been found. Though a complete characterization of the universal information inequality remains elusive, we know that exactly the same classes of linear inequalities are valid for Shannon's entropy and for Kolmogorov complexity (this result was proven in the PhD thesis of the author). The established equivalence suggests that these inequalities (common for

the probabilistic and the algorithmic versions of information theory) express fundamental properties of the notion of "information," and the further research of information inequalities seems to be a promising direction.

In a joint paper with Konstantin and Yury Makarychev, and Nikolai Vereshchagin we proposed a new method of deriving information inequalities. This technique uses the connection between the information inequalities and the idea of "extracting" the mutual information (which goes back to the works by R. Ahlswede, P. Gács, and J. Körner in 1970s). This type of argument helped to reveal the intuitive meaning of the Zhang–Yeung inequality and to discover an infinite series of new information inequalities.

Since late 1990s, the researchers discovered several examples of so-called *constraint information inequalities* — linear inequalities for Shannon's entropy that are valid for the distribution that match some linear conditions for the values of entropies. In a series of joint works with Tarik Kaced we showed that some of these inequalities are, so to say, substantially conditional. This means that they cannot be obtained as a projection of any conventional (unconditional) information inequality, known or yet unknown. We established the connection of the conditional inequalities with the geometry of the cone of the unconditional inequalities of entropies. Loosely speaking, one single constraint inequality for a quadruple of random variables corresponds to infinitely many conventional (unconditional) linear inequalities for entropies. The existence of such a constraint inequality explains why the cone of unconditional inequalities for n-tuples of random variables (with $n \geq 4$) is not polyhedral (a result proven earlier by F. Matúš). We studied the counterparts of the constraint information inequalities in the framework of Kolmogorov complexity. It turned out that some of these inequalities have a valid homologue for Kolmogorov complexity, while others do not. Apparently, such a divergence between the properties of Shannon's entropy and Kolmogorov complexity deserves closer attention. By exploring these exceptional constraint inequalities (that are valid in Shannon's but not in Kolmogorov's framework), we found interesting applications of information theory to combinatorics and communication complexity, as described below.

Materialization of mutual information and conditional complexities. For each pair of strings x, y, we consider the conditional complexities $C(x|y)$ and $C(y|x)$ (the lengths of the shortest programs that transform x to y and y to x) and the value of the mutual information $I(x : y)$ (that can be defined in a symmetric form as $C(x)+C(y)-C(x,y)$). There is a long-standing question on a more explicit and operational interpretation of these information quantities. A version of these questions for Shannon's entropies was extensively studied in 1970s (e.g., the Slepian–Wolf theorem gives an operational characterization of the conditional entropy, and Gács and Körner showed that the mutual information in general cannot be materialized). In the algorithmic framework these questions become popular in 1990s.

In the late 1990s, An. Muchnik proposed a nice interpretation of the value of $C(x|y)$. He showed that this quantity can be understood not only as the length of the shortest programs that transform one string to another but also as the optimal "fingerprint" (or a "hash value"), which (i) can be extracted directly from x, and (ii) pro-

vides enough information to reconstruct x given y. In other words, the quantity of $C(x|y)$ can be "extracted" directly from x, with almost no information on y.

In a joint work with Daniil Musatov and Alexander Shen we presented two new proofs of this theorem. One of these constructions is based on a so-called on-line matching algorithm for bipartite graphs. The second one, based on extractors, can be generalized to prove a version of Muchnik's theorem for space-bounded Kolmogorov complexity. We proved another version of Muchnik's theorem for a resource-bounded variant of Kolmogorov complexity based on Arthur–Merlin protocols. In subsequent works by M. Zimand, a time-bounded version of Muchnik's theorem was established (Zimand showed that the fingerprint can computed from x by a randomized algorithm very efficiently, and only the reconstruction of x from y and the fingerprint is time consuming).

It is worth mentioning that the techniques used in proofs of resource-bounded versions of Muchnik's theorem involve different versions of pseudo-random graphs (extractors, expanders). Quite interestingly, the technique of pseudo-random fingerprints developed in the context of a resource-bounded version of Muchnik's theorem has found an application in a fairly different area. We used it to construct a data structure (a bit probe scheme) that stores a finite set so that membership queries can be answered very efficiently, by retrieving only one bit from the database.

The problem of materialization of mutual information of strings seems to be trickier than a materialization of conditional complexity. The naive intuition suggests that the mutual information between x and y represents the quantity of data shared by two strings. So researchers have explored the extent to which mutual information can be extracted from x and y, in one way or another. P. Gács and J. Körner, in this context, introduced the term *common information*. Very informally, a string z is common information extracted from x and y, if z can be "easily" computed from x, and also from y, i.e., $C(z|x) \approx 0$ and $C(z|y) \approx 0$. It can be shown that common information is upper bounded by mutual information (up to logarithmic precision). Gács and Körner (1973) showed that from some x and y the common information is much smaller than the mutual information. Moreover, the possibility of extracting the mutual information does not depend solely on the basic complexity measures involving x and y; there exist pairs (x_1, y_1) and (x_2, y_2) that have the same complexity profile, though for (x_1, y_1) the common information and the mutual information are equal, whereas for (x_2, y_2) they are drastically different. Similar negative results (the common information can be much less than the mutual information, even with a rather loose threshold for the values of $C(z|x)$ and $C(z|y)$) were established by a group of researchers at Moscow University in 1990s (Muchnik, Shen, Vereshchagin, and others; several results of this type were presented in the PhD thesis of the author).

The problem of extracting the mutual information can be restated for a triple of strings. We say that the mutual information shared by strings x, y, z can be completely materialized if there exists a string w such that x, y, and z are independent given z, and z is simple conditional on each of the strings x, y, z. We showed that the question of extracting the mutual information for a triple of strings has a surprising positive answer: the mutual information shared by strings x, y, and z completely

materialized, if and only if, the conditional mutual informations $I(x:y|z)$, $I(x:z|y)$, and $I(y:z|x)$ are negligibly small. The proof of this result heavily depends on the new non-Shannon-type information inequalities mentioned above.

Combinatorial applications. The idea of applying Shannon's entropy in combinatorial proofs is by no means new. A textbook example of such an argument is the combinatorial Shearer lemma, which is *per se* a translation of an inequality for Shannon's entropy in the language of Hartley's information (i.e., in terms of sizes of finite sets). However, the combinatorial applications of information-theoretic arguments typically look like a smart *ad hoc* trick. At the same time, the idea by Kolmogorov of the parallelism between three approaches to the information theory suggests that every information inequality (which is valid both for Shannon's entropy and for the Kolmogorov complexity) might be translated to an equivalent combinatorial statement. In a joint work with A. Shen and N. Vereshchagin we tried to establish this connection formally. We proved that every valid linear inequality for the Kolmogorov complexity or Shannon's entropy can be translated into an equivalent combinatorial statement. Shearer's combinatorial inequality can be considered as a special case for this general translation. Later, another version of a combinatorial interpretation of information inequalities was suggested by A. Shen and N. Vereshchagin in a joint work with I. Newman and G. Tárdos.

For the constraint information inequalities we still do not have a similar general scheme translating every inequality to an equivalent combinatorial statement. However, for some specific constraint information inequality we found some combinatorial applications. The resulting combinatorial statements can be naturally stated in terms of edge coloring for bipartite graphs (a joint work with T. Kaced and N. Vereshchagin). From this perspective, some constraint information inequalities can be understood as lower bounds for biclique cover of graphs. This technique was used in proofs of negative results in communication complexity, as described below.

An operational characterization of the mutual information. In a joint work with Marius Zimand we revisited the question of materializing the mutual information in algorithmic information theory and proposed a sort of "operational characterization" of the value of mutual information.

We show that the mutual information of a pair of strings x and y is equal to the size of the longest shared secret key that two parties, one having x and the other one having y, can establish via a probabilistic protocol with interaction on a public channel. This result is an extension to the case of $n > 2$ parties. In this work we simultaneously employed various results of algorithmic information quantities mentioned above: an operational characterization of conditional complexity helped to prove the positive result and set up a communication protocol that achieves the optimal size of the shared secret key; a constraint information inequality was used to prove the negative results (the size of the common secret key cannot be larger than the mutual information between the inputs strings); and lastly, the negative results on extracting the common information was used to estimate the optimal communication complexity of this communication problem (if the communication complexity

drops below the established threshold then only negligibly small secret keys can be obtained).

3. Shifts of finite type: local rules and information density

The other part of this research is devoted to multi-dimensional shifts. The *shifts* are the translation invariant and topologically closed sets of configurations of a Cantor space (more precisely, the translation invariant and topologically closed sets of configurations on $\mathbb{Z}^d$ over a finite alphabet). Every shift can be uniquely determined by some set of *forbidden patterns* (so that a configuration belongs to the shift, if and only if, it does not involve any forbidden finite pattern). The *shifts of finite type* (SFT) are the shifts defined by forbidding a finite number of patterns.

The shifts of finite type provide a playground to study the appearance of non-trivial global phenomena from simple local rules. The SFT play a prominent role in several areas such as mathematical logic, language theory, computational complexity, symbolic dynamics and even mathematical physics (where the SFT are used as a simplistic model of crystalline and quasicrystalline structures).

Many profound results concerning SFT depend heavily on embedding a computation in the geometric structure of a shift. We proposed a new technique of embedding a Turing machine in an SFT. This technique is referred to as *self-simulating* or *fixed-point tilings*. It combines the idea of geometric self-similarity with the idea of a self-referential program.

The employed technique of self-referential programming is similar to the idea of Kleene's recursion theorem. Even closer analogues to our technique are the constructions of self-reproducing automata by J. von Neumann and especially the self-correcting cellular automata by P. Gács. It turned out that the idea of a self-referential program matches perfectly the idea of a self-similar geometric configuration that reproduces similar patterns on different scales. Using this technique we give new proofs of some classic theorems of symbolic dynamics and then obtain several new results. Below, we briefly describe the main points of this research.

Simple local rules imply complex global behavior. For every $d \geq 2$ there exists a non-empty SFT of dimension d where all configurations are aperiodic (R. Berger, 1966). The very existence of aperiodic SFT is non-trivial and has interesting implications. What is even more important, aperiodic SFT are used as basic frameworks for constructions with various non-trivial properties.

The construction suggested by Berger was quite technical. Later, several simpler proofs of this result were suggested. All known constructions of aperiodic SFT combine in different proportions geometric and algorithmic ideas. We propose a construction (based on "self-simulating tilings") which is in some sense extremal: it is almost purely algorithmic, with only very few geometric tricks. This approach does not help minimize the size of the alphabet or the number of local constraints in an SFT. However, it is pretty flexible and admits various generalizations and extensions, which is crucial for applications.

We started with several new proofs of previously known results. One of the first applications of the new technique was a simplified proof of the classic result by

W. Hanf and D. Myers (1974): for every $d \geq 2$ there exists a non-empty SFT where all configurations are uncomputable. This technique was extended then to construct an SFT with a maximal possible "density of information." More precisely, for every $d \geq 2$ we constructed an SFT of dimension d, where for all configurations each pattern of size $n \times n \times \cdots \times n$ has Kolmogorov complexity $\Omega(n^{d-1})$. (An SFT with high information density was originally obtained by B. Durand, L. Levin, and A. Shen (2001) with a technique based on Robinson's tilings.)

An extension of the result concerning SFT with high Kolmogorov complexity mentioned above led us to a better understanding of the projective subdynamics of multidimensional SFT. We showed that every effective shift on $\mathbb{Z}^d$ can be represented as a projection of the subdynamics in an SFT on $\mathbb{Z}^{d+1}$, which answers a question raised by M. Hochman (2009). (The latter result was independently proven by N. Aubrun and M. Sablik (2013) with a different technique.)

Quasiperiodic and minimal SFT. A. Ballier and N. Ollinger (2009) showed that there exists an SFT where each configuration is at once aperiodic and quasiperiodic (this SFT is even minimal). We suggested a new proof of this result, based again on self-simulating tilings. In this noteworthy example we employed an intrinsically algorithmic technique of self-referential programs to prove a theorem with no algorithmic notions in its assertion.

We also proved that there exists an SFT where each configuration is at once non-computable and quasiperiodic. Moreover, we obtained a characterization of the classes of Turing degrees that can by represented by the configurations of a quasiperiodic SFT. To the best of our knowledge, the only known proofs of these results use the technique of self-simulating tilings.

Robust (fault-tolerant) SFT. We show that several results mentioned above (aperiodic SFT, strongly aperiodic SFT, SFT with patterns with high Kolmogorov complexity, and so on) remain true for a more general type of objects – for so called *faulty tilings*, i.e., for configurations where local constraints can be violated at a sparse set of randomly chosen "errors." The technique of self-simulating tilings is, until now, the only known method to prove this type of result on robust SFT. In some of these results we observe again that a heavily algorithmic technique can be useful in proofs of mathematical theorems with no algorithmic issues in their statements.

Concluding remarks. The technique of self-simulating tilings *à la* Kleene appeared in our series of joint papers with Bruno Durand and Alexander Shen. Our work was largely inspired by the construction of reliable cellular automata with self-organization suggested by P. Gács. In recent years, the technique of self-simulating tilings was redeveloped by L.B. Westrick and C. Zinoviadis.

5. Our contribution: new techniques and an interplay between different frameworks

In the first half of this manuscript we study the fundamental notion of randomness. We discuss various versions of information measures in the algorithmic, probabilis-

tic, and combinatorial frameworks. Our studies are motivated by parallels between alternative approaches to the notion of information developed in different communities. In this research we employ similar arguments in heterogeneous contexts, implementing pretty close intuitive ideas with different formal tools. We suggest several new techniques (e.g., the method of "clones" in Kolmogorov complexity, the method of constraint information inequalities, and so on); but the principal contribution of this part of our work consists in adapting the existing techniques of information theory (information inequalities, extraction of common information, pseudo-random graphs, and so on) in new contexts.

Another part of the manuscript is built around one specific technical idea — a construction of self-simulating tilings. We study different aspects of this construction, from the basic generic scheme to various extensions and enhancements, and show several applications where this technique proves to be efficient. The concerned results share not intuitive insights but a similar technical implementation.

The questions discussed in these two parts of the manuscript are pretty different. Nevertheless, we emphasize that they are connected by one fundamental concept — by Kolmogorov complexity, which gives the measure of information in a finite object in the first part and the measure of information density in an infinite configuration in the second part of the manuscript. At the end of the manuscript we outline a prospective research project. In this project we sketch the directions (based on resource-bounded Kolmogorov complexity) that can eventually bring these fields of research closer together.

Organization of the Manuscript

The scientific content of the manuscript is presented in Chapters 1 and 2. In Chapter 1, we study the fundamental complexity measures of algorithmic information theory: plain and conditional Kolmogorov complexity, mutual information, conditional mutual information, and so on. We discuss the universal information inequalities, different aspects of materialization of the mutual information, with a focus on the interplay between algorithmic and probabilistic versions of information theory, and on their applications in combinatorics. In Chapter 2, we study the problems of symbolic dynamics: shifts of finite type and Wang tilings, their algorithmic, topological, and combinatorial properties. Kolmogorov complexity is employed in this chapter as a measure of the density of information, which appears interesting from the algorithmic and combinatorial perspectives. The principal tool in Chapter 2 is the technique of self-simulating tilings.

Chapter 1 is organized raditionally: in each section, we discuss a particular group of results, explaining the context around and the driving motivations behind, and the principal ideas employed in the proofs of the concerned theorems. In most cases, the technical proofs are deferred to the full versions of the original papers. (Some of these papers are attached in the appendices of the manuscript.) However, in several cases we sketch in the body of the manuscript the proofs of weaker versions of our main results. We do it when we believe that a minor loss of generality simplifies the argument significantly, and this simplification makes the methods used in the proof more transparent.

In what follows we outline the content of Chapter 1. In Sections 1.1–1.5 we discuss the context of our research (including several results from the PhD thesis of the author). The main contribution is presented in Sections 1.6–1.14.

- Section 1.1: the basic definitions of algorithmic complexity.
- Section 1.2: measures of information by Shannon and by Hartley.
- Section 1.3: the principal information quantities in algorithmic and probabilistic versions of information theory.
- Section 1.4: the universal information inequalities (the general setting and a survey of known results).
- Section 1.5: the notion of (α, β)-stochasticity.
- Section 1.6: Muchnik's theorem on the optimal conditional description and its variations (on a joint work with D. Musatov and A. Shen, [c8], [j8]).
- Section 1.7: efficient bit-probe schemes, on [j5].
- Section 1.8: the common information of a pair: positive and negative results (on a joint work with An. Muchnik, [c10], [j9]).
- Section 1.9: algorithmic version of the Ahlswede-Körner lemma and the inference of non-Shannon-type information inequalities (on joint works with

K. Makarychev, Yu. Makarychev, N. Vereshchagin, [j13], and An. Muchnik, [j9]).

- Section 1.10: application of non-Shannon-type information inequalities to extracting the common information of a triple, on [c13], [j12].
- Section 1.11: constraint information inequalities (on a joint work with T. Kaced, [c20], [c17], [j6]).
- Section 1.12: combinatorial interpretation of (unconditional) information inequalities (on a joint work with A. Shen and N. Vereshchagin, [c14], [j14]).
- Section 1.13: combinatorial application of constraint information inequalities. (on a joint work with T. Kaced and N. Vereshchagin, [j2]).
- Section 1.14: common secret key agreement in the framework of Kolmogorov complexity (on a joint work with M. Zimand, [c1], [e1]).

To the experts in the field, we suggest starting the reading with the very last section of the chapter (Section 1.14). This section sets out recent results on a secret key agreement in the context of algorithmic information theory. Quite noteworthy, this work combines the techniques from most of the previous sections, justifying a posteriori the former research on extracting common information, information inequalities, and their combinatorial interpretation.

Chapter 2 is organized very differently. In this chapter, we use the proofs based on hierarchical self-simulating tilings, and in these arguments, it is hard to separate the core technique and the features specific to a particular application. In every new result, we cannot only cite the statements of several previously known theorem, but we also have to re-explain the proofs of the former results and to reemploy once again the entire constructions of these proofs (adding several new twists and embedding new gadgets in the same general scheme). This makes the proofs long and somewhat cumbersome, and every original paper begins with reproducing the same general technical framework. Instead of providing texts of original papers with vastly overlapping technical sections, we present in Chapter 2 a self-contained exposition of the proofs of our principal results. We start with a general perspective of this technique and show how it applies to several classic results. Then, in each succeeding section, we explain how to adjust and extend the generic construction to prove one or another particular theorem.

We outline the content of sections of Chapter 2:

- Section 2.1: discussion of the context and of the principal motivations in the field.
- Section 2.2: the basic definitions used in symbolic dynamics.
- Section 2.3: the core technique of self-simulating tilings that appeared in [c9], [c7], [j10], [c19], [j7].
- Section 2.4: applications to classical results (SFTs with non-computable configurations), on [j7].

- Section 2.5: quasiperiodic constructions in the framework of self-simulating SFTs, on [c4].
- Section 2.6: combining quasiperiodicity and non-computability, [c4].
- Section 2.7: a characterization of one-dimensional subdynamics of two-dimensional SFTs, on [j7].
- Section 2.8: SFTs with high information density and strongly aperiodic SFTs, on [c7], [j7].
- Section 2.9: on subdynamics of quasiperiodic SFTs, on [c2].
- Section 2.10: faulty tilings and robust SFTs, on [c9], [c7], [j7].

The results of this chapter were published in a series of joint papers with Bruno Durand and Alexander Shen, [c9], [c7], [j10], [j7], [c4], [c2]. We follow mostly the exposition from the journal version of [j7] and from the preprint [e2] (though we do not reproduce them literally). We use in this manuscript several figures from [e2].

Chapter 1 and Chapter 2 can be read independently. At the end of the manuscript we summarize open problems and outline the future work.

Notation

In the manuscript we use the following standard notation:

- $\#S$ stands for the cardinality of a finite set S;
- for a string (a word over a finite alphabet) x we denote by $|x|$ the length of x (the number of letters in the word);
- given a tuple of strings $(x_1,\dots,x_k)$, for every set of indices $V=\{i_1,\dots,i_s\}$ (where $1\le i_1<\dots<i_s\le k$) we denote by x_V the tuple $(x_{i_1},\dots,x_{i_s})$; a similar notation is used for tuples of random variables;
- we abbreviate $\mathscr{L}_1\le\mathscr{L}_2+O(\log n)$ to $\mathscr{L}_1\le^+\mathscr{L}_2$, where the parameter n is clear from the context (n is typically the Kolmogorov complexity of the strings involved in terms $\mathscr{L}_1$ and $\mathscr{L}_2$); similarly, we use the notation $\mathscr{L}_1\ge^+\mathscr{L}_2$ and $\mathscr{L}_1=^+\mathscr{L}_2$;
- all logarithms in the manuscript are to base 2;
- $\mathrm{C}(x)$ denotes the Kolmogorov complexity of a string x;
- $H(X)$ denotes Shannon's entropy of a random variable X;
- we keep the usual notation $I(\cdot)$ for the mutual information in the frameworks of Kolmogorov complexity and Shannon's entropy (it is clear from the context which version of the mutual information we discuss).

Chapter 1

Algorithmic Information Theory: Fundamental Complexity Quantities and Information Measures

1.1 The basic definitions of algorithmic information theory

As we have mentioned in the introduction, the notion of algorithmic complexity was introduced in the 1960s in works by A. Kolmogorov, R. Solomonoff, and G. Chaitin (in slightly different forms, with somewhat different motivations). The seminal paper by Kolmogorov, [3], is for us the most important source of motivations and fundamental ideas. We start with the very basic definitions of the theory of algorithmic descriptive complexity.

Kolmogorov's measure of algorithmic complexity. Let us recall the main definitions of algorithmic complexity (usually called nowadays Kolmogorov complexity).

Definition 1. *Let U be a (partial) computable function of two arguments. The conditional complexity of x given y (with respect to the description method U) is*

$$\mathrm{C}_U(x \mid y) := \min\{\, |p| \,:\, U(p,y) = x \,\}.$$

If there is no p such that $U(p,y) = x$, we assume that $\mathrm{C}_U(x \mid y) = \infty$.

The intuition behind this definition is simple: U is understood as a "description method" or a programming language (or, more precisely, an interpreter of this programming language); p is a program that takes y as an input and returns x as an

output. The complexity of x given y is the length of (one of) the shortest programs p that transforms y into x.

The obvious problem with this definition is its dependence on U: for different "programming languages" U the resulting complexity measures $\mathrm{C}_U(x \mid y)$ can be drastically different. Apparently, we are interested in more economical description methods. The next definition helps compare different instantiation of the complexity $\mathrm{C}_U(x \mid y)$ with each other.

Definition 2. *Let U and V be a computable functions of two arguments. We say that U is* not worse *than V as a description method, if there exists a number c such that for all strings x, y*

$$\mathrm{C}_U(x \mid y) \le \mathrm{C}_V(x \mid y) + c$$

(the complexity in the sense of U is asymptotically not greater than the complexity in the sense of V).

The theory of Kolmogorov complexity is possible due to the following result.

Theorem 1 (Kolmogorov, [3])**.** *There exists a computable function of two arguments U that is not worse as a description method (in the sense of Definition 2) than any other V.*

A computable function U that is not worse as a description method than any other V is called an *optimal description method*. Such an optimal description method is not unique. If U_1 and U_2 are two optimal description methods, then there exists a number c such that for all strings x, y

$$|\mathrm{C}_{U_1}(x \mid y) - \mathrm{C}_{U_2}(x \mid y)| \le c,$$

so the choice of the optimal description method has little importance. We fix once and forever an arbitrary optimal U (whose existence is guaranteed by Theorem 1); in what follows we omit the subscript U and denote $\mathrm{C}_U(x \mid y)$ by $\mathrm{C}(x \mid y)$. The value $\mathrm{C}(x \mid y)$ is called *Kolmogorov complexity* of x conditional on y.

We denote by $\mathrm{C}(x)$ (the *unconditional Kolmogorov complexity* of x) the value of $\mathrm{C}(x \mid \Lambda)$, i.e., the complexity of x given the empty word Λ.

In a similar way, we define Kolmogorov complexity in terms of programs with bounded resource (space and time). In this definition, we should specify explicitly the computational model. We use the conventional model of computations — Turing machines.

Definition 3. *Let U be a Turing machine; we define the conditional Kolmogorov complexity $\mathrm{C}_U^{t,s}(x \mid y)$ as the length of the shortest p such that $U(p, x)$ produces y in at most t steps using at most s cells on the tape.*

It is known that there exists an *optimal description method U* in the following sense: for every Turing machine V

$$\mathrm{C}_U^{\mathrm{poly}(t),O(s)}(x \mid y) \le \mathrm{C}_V^{t,s}(x \mid y) + O(1).$$

For multi-tape Turing machines a slightly stronger statement can be proven:

Theorem 2 (see [76], with the simulation technique from [5])**.** *There exists an* optimal description method (*multi-tape Turing machine*) U *in the following sense: for every multi-tape Turing machine* V *there exists a constant* c *such that for all strings* x, y

$$\mathrm{C}_U^{ct\log t,cs}(x \mid y) \le \mathrm{C}_V^{t,s}(x \mid y) + c.$$

We fix such a method U, and in the sequel use for the resource-bounded version of Kolmogorov complexity the notation $\mathrm{C}^{t,s}$ instead of $\mathrm{C}_U^{t,s}$. We use the notation $\mathrm{C}^{t,\infty}$ and $\mathrm{C}^{\infty,s}$ if only time or only space available to the algorithm are bounded.

In algorithmic information theory, we want to handle the Kolmogorov complexities of compound objects, e.g., pairs of strings, triples of strings, etc. To this end, we fix a computable enumeration of tuple of strings, i.e., a computable injective mapping

$$\langle x_1, \dots, x_k \rangle \mapsto code(x_1, \dots, x_k)$$

that assigns to every k-tuple of strings its *code* (which is a binary string itself). Kolmogorov complexity of a k-tuple of strings $\langle y_1, \dots, y_k \rangle$ (possibly given another m-tuple of strings $\langle x_1, \dots, x_m \rangle$) is defined as the Kolmogorov complexity of the code of the given tuple (respectively, given the code of another tuple). To simplify the notation, we write

$$\mathrm{C}(y_1, \dots, y_k) \text{ and } \mathrm{C}(y_1, \dots, y_k \mid x_1, \dots, x_m)$$

instead of

$$\mathrm{C}(code(y_1, \dots, y_k)) \text{ and } \mathrm{C}(code(y_1, \dots, y_k) \mid code(x_1, \dots, x_m))$$

respectively. The change of the encoding affects the value of Kolmogorov complexity by only an $O(1)$ terms, so we do not have to specify our choice of the computable enumeration of tuples.

For a finite set S (of strings or tuples of strings) we define its Kolmogorov complexity as the Kolmogorov complexity of the list of all elements of S in alphabetical order.

In this manuscript we do not provide an exhaustive exposition of the basic properties of Kolmogorov complexities. For an extensive survey on this theory, we refer the reader to [8], [27], [65], [76], [99]. In this introduction, we only recall the basic properties of Kolmogorov complexity that are crucial for algorithmic information theory.

Remarks on incompressible words. From a simple counting argument, it follows that for every n there is a binary string x of length n whose complexity is at least n (i.e., $\mathrm{C}(x) \ge n$). Such a string is called *incompressible*. Similarly, for every integer $c > 0$ there are at most 2^{n-c} strings of length n whose complexity is less than $n - c$. Thus, most strings of length n are almost incompressible (for all strings of length n except for a fraction of size 2^{-c} their Kolmogorov complexity is not less than $n - c$).

Let us mention another (much more involved) fact concerning infinite sequences: there exists a bi-infinite binary sequence where all factors of length n have Kolmogorov complexity $\Omega(n)$. This result is known as Levin's lemma:

Lemma 1. *(a) [one-dimensional version] For every $c < 1$ there exists a bi-infinite binary sequence $(\dots x_{-1}x_0x_1x_2\dots)$ such that for all large enough n and for all k*

$$\mathrm{C}(x_k x_{k+1} x_{k+2} \dots x_{k+n-1}) > cn.$$

(b) [multi-dimensional version] for every integer number $d > 1$ and for every real number $c < 1$ there exists a d-dimensional configuration

$$f : \mathbb{Z}^d \to \{0,1\}$$

such that for all large enough n the Kolmogorov complexity of every $n \times n \times \dots \times n$ pattern in this configuration is greater than cn^d.

Observe that a randomly chosen infinite sequence (or a randomly chosen d-dimensional configuration) with probability 1 contains all possible finite patterns and therefore does not satisfy Lemma 1. In particular, this property does not hold for Martin-Löf random sequences (see the definition of infinite algorithmically random objects in [6] and in the textbooks [76], [99]). Lemma 1 can be proven with the chain rule for the prefix-free version of Kolmogorov complexity or with the Local Lovász Lemma, see [68], [74].

Basic notions of algorithmic information theory. We use the standard notation for the mutual information:

$$I(x : y) := \mathrm{C}(x) - \mathrm{C}(x \mid y)$$

and its conditional version

$$I(x : y \mid z) := \mathrm{C}(x \mid z) - \mathrm{C}(x \mid y, z).$$

Now we proceed with the algorithmic version of the "chain rule." This theorem was historically one of the first nontrivial results on Kolmogorov complexity:

Theorem 3 (Kolmogorov–Levin, [8])**.** *There exist constants c_1 and c_2 such that for all strings x, y*

$$|\mathrm{C}(x, y) - \mathrm{C}(x) - \mathrm{C}(y \mid x)| \le c_1 \log \mathrm{C}(x, y) + c_2. \tag{1.1}$$

The logarithmic term in (1.1) cannot be avoided: the gap between $\mathrm{C}(x, y)$ and $\mathrm{C}(x) + \mathrm{C}(y \mid x)$ can be indeed of size $\Theta(\log(\mathrm{C}(x, y)))$, for arbitrarily large strings x and y. Note that (1.1) can be reformulated in our notation as

$$\mathrm{C}(x, y) =^+ \mathrm{C}(x) + \mathrm{C}(y \mid x). \tag{1.2}$$

Theorem 3 implies the symmetry of the mutual information:

Corollary 1. *The mutual information is symmetric (up to a logarithmic term),*

$$I(x : y) =^+ I(y : x) =^+ \mathrm{C}(x) + \mathrm{C}(y) - \mathrm{C}(x, y).$$

A similar statements holds for the conditional mutual information,

$$I(x : y \mid z) =^+ I(y : x \mid z) =^+ \mathrm{C}(x, z) + \mathrm{C}(y, z) - \mathrm{C}(x, y, z) - \mathrm{C}(z).$$

1.2 Non Algorithmic Measures of Information: Shannon's and Hartley's Approaches.

The original paper by Kolmogorov [3] was entitled "Three approaches to the quantitative definition of information." One of these approaches was algorithmic, while two others were the probabilistic and the combinatorial ones. Kolmogorov highlighted the parallels and close connections between these three formalisms.

In the probabilistic approach, which goes back to Claude Shannon, [2], the object that contains "information" is a random variable (or, more precisely, a probabilistic distribution). The measure of information in a distribution is defined as Shannon's entropy of this distribution. For a discrete distribution X with a finite range

values of X:	a_1	a_2	$\dots$	a_k
probabilities:	p_1	p_2	$\dots$	p_k

(with non-negative probabilities p_i that sum to 1) its entropy is defined as

$$H(X) := \sum_{i=1}^{k} p_i \log \frac{1}{p_i},$$

with the usual convention $0 \cdot \log \frac{1}{0} = 0$.

Shannon's entropy of X provides the asymptotically optimal compression ratio for a flow of letters over an alphabet $a_1, \dots, a_k$ with frequencies $p_1, \dots, p_k$. Thus, similar to the algorithmic complexity, Shannon's approach measures a sort of optimal compression rate. The difference between these approaches is that Shannon's measure of information focuses only on the frequencies of letters, while the Kolmogorov complexity deals with all types of regularities.

For jointly distributed random variables X and Y, for each value b_j of Y we can consider the corresponding conditional distribution on X and the value of Shannon's entropy for this conditional distribution, denoted $H(X \mid Y = b_j)$. The weighted average of these values is called the *conditional entropy* of X given Y,

$$H(X \mid Y) := \sum_j \text{prob}[Y = b_j] \cdot H(X \mid Y = b_j).$$

Thus, for a pair of jointly distributed random variables (X, Y) we have the following standard information quantities: the entropy of the entire distribution $H(X, Y)$, the entropies of the marginal distributions $H(X)$ and $H(Y)$, and the conditional entropies $H(X \mid Y)$ and $H(Y \mid X)$. A version of the "chain rule" for Shannon's entropy can be easily deduced from the basic definitions:

$$H(X, Y) = H(X) + H(X \mid Y). \tag{1.3}$$

It follows immediately that the mutual information defined as

$$I(X : Y) := H(Y) - H(Y \mid X)$$

is symmetric:

$$I(X : Y) = I(Y : X) = H(X) + H(Y) - H(X, Y). \tag{1.4}$$

Given a jointly distributed triple of random variables (X, Y, Z), we can define the conditional mutual information

$$I(X : Y \mid Z) := H(Y \mid Z) - H(Y \mid X, Z).$$

This quantity also enjoys the property of symmetry:

$$\begin{aligned} I(X : Y \mid Z) = I(Y : X \mid Z) &= H(X \mid Z) + H(Y \mid Z) - H(X, Y \mid Z) \\ &= H(X, Z) + H(Y, Z) - H(X, Y, Z) - H(Z) \end{aligned} \tag{1.5}$$

For a detailed introduction to Shannon's information theory, we refer the reader to the classic textbooks [65], [85].

We can see now that even syntactically the basic properties of the Kolmogorov complexity and Shannon's entropy look very similar: we have similar expressions of the chain rule, similar properties of symmetry of the mutual information, and so on. In the present manuscript, we address this parallelism in several different contexts. In some cases we establish a formal equivalence: the properties of algorithmic information theory can be translated into formally equivalent properties of Shannon's information theory, and visa-versa. Thus, in these cases, the technique from one theory can be re-employed to prove statements in the other one. In other cases we will use this parallelism as a heuristics: knowing a statement from one incarnation of information theory we can conjecture plausible homologous statements in the other one, even though there is no formal correspondence between these statements. Furthermore, in some cases we observe a discrepancy between the algorithmic and the probabilistic versions of information theory; and while exploring the profound reasons explaining this discrepancy we reveal particularly interesting properties of both Kolmogorov complexity and Shannon's entropy.

In conclusion of this section, we mention the third approach to the quantitative definition of information — the combinatorial one. It goes back to Ralph Hartley, [1], who defined the information contained in a finite set as a logarithm of the cardinality of this set. On the one hand, this idea is very natural, since we need $\lceil \log \# A \rceil$ binary digits to specify an element in a set A. On the other hand, this definition seems to be very naive and simplistic. Nevertheless, the combinatorial approach turns out to be far from trivial. Even very elementary questions on Hartley's information lead to interesting and non self-evident combinatorial results. Actually, the interplay between Hartley's, Shannon's, and Kolmogorov's approaches provides a wide range of applications of Shannon's entropy and Kolmogorov complexity. We discuss several combinatorial applications of information theory in Section 1.12.

1.3 Information Quantities for a k-tuples of Correlated Objects

For an individual bit string x we have only the value of its Kolmogorov complexity $\mathrm{C}(x)$, and this single quantity represents the most fundamental properties of x from the perspective of algorithmic information theory. For a pair of strings, x and y,

we have several different meaningful information quantities. First of all, we have the Kolmogorov complexities of both strings, $C(x)$ and $C(y)$, and the Kolmogorov complexity of the pair, $C(x,y)$. Besides, we have two conditional complexities $C(x \mid y)$ and $C(y \mid x)$, and the values of the mutual information $I(x:y)$ and $I(y:x)$. There are several well-known correlations between these quantities. From Theorem 3 it follows that

$$\begin{aligned} C(x,y) &=^+ C(x) + C(y \mid x), \\ C(x,y) &=^+ C(y) + C(x \mid y), \\ I(x:y) &=^+ C(x) + C(y) - C(x,y), \\ I(x:y) &=^+ I(y:x), \end{aligned}$$

where all equations hold "up to a logarithmic additive term," i.e., up to the term $O(\log C(x,y))$. It is remarkable that the logarithmic residue terms cannot be eliminated from these relations (see [8]). In what follows we ignore the accuracy issue and always consider relations between information quantities up to an unavoidable logarithmic term (which is assumed to be negligibly small with respect to the complexity of the strings involved).

We observe that all information quantities for x and y listed above can be expressed (up to a logarithmic term) as linear combinations of three "primary" quantities. The choice of the triple of primary quantities (like the choice of a coordinate system in a three-dimensional space) is not unique. One standard approach is to choose as the "basic coordinates" the values of unconditional complexities $C(x)$, $C(y)$, and $C(x,y)$. Then the other quantities for the triple can be expressed as

$$\begin{aligned} C(x \mid y) &=^+ C(x,y) - C(y), \\ C(y \mid x) &=^+ C(x,y) - C(x), \\ I(x:y) &=^+ C(x) + C(y) - C(x,y), \\ I(y:x) &=^+ C(x) + C(y) - C(x,y). \end{aligned}$$

In some cases it is more convenient to chose other primary quantities: the values of the conditional complexities $C(x \mid y)$ and $C(y \mid x)$, and the mutual information $I(x:y)$. Then the other quantities can be represented as

$$\begin{aligned} C(x) &=^+ C(x \mid y) + I(x:y), \\ C(y) &=^+ C(y \mid x) + I(x:y), \\ C(x,y) &=^+ C(x \mid y) + C(y \mid x) + I(x:y), \\ I(y:x) &=^+ I(y:x). \end{aligned}$$

These simple relations suggest a geometric representation of all standard information quantities: we can visualize the "complexity profile" of a pair of strings (x,y) by the Venn-like diagram, as shown in Fig. 1.1. In this picture, the area covered by the left circle corresponds to the value of $C(x)$, and the area covered by the right circle corresponds to the value of $C(y)$; the area of the union of both circles corresponds to the value of $C(x,y)$. Accordingly, the complement of one circle to another corresponds to the conditional complexity, i.e., in this picture the size of the red area represents $C(x \mid y)$ and the size of the blue area represents $C(x \mid y)$. The size of the intersection of the circles (the violet area in the picture) represents the value of $I(x:y) =^+ I(y:x)$.

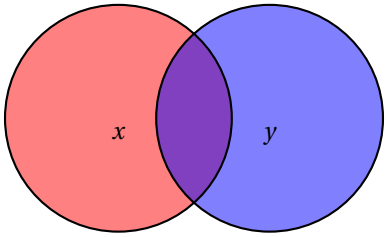

Figure 1.1: The diagram representing the information quantities of a pair (x, y).

All quantities shown in Fig. 1.1 (the "atomic" quantities $\mathrm{C}(x \mid y)$, $\mathrm{C}(y \mid x)$, $I(x : y)$ and their combinations) are non-negative for all pairs (x, y) (as usual, up to a logarithmic term). The non-negativity of $\mathrm{C}(x \mid y)$ and $\mathrm{C}(y \mid x)$ is trivial; the non-negativity of the mutual information is a reformulation of the well-known inequality

$$\mathrm{C}(x, y) \leq^{+} \mathrm{C}(x) + \mathrm{C}(y)$$

(which claims that we can combine a description of x and a description of y in a joint description of the pair (x, y), and the joining of two descriptions requires only a logarithmic overhead).

Let us extend the observations made above and adapt this geometric representation to a triple of binary strings. Given a triple of binary strings x, y, z we deal with a bunch of information quantities. First of all, we have the values of Kolmogorov complexity for each string, for each pair of strings, and for the entire triple,

$$\mathrm{C}(x), \mathrm{C}(y), \mathrm{C}(z), \mathrm{C}(x, y), \mathrm{C}(x, z), \mathrm{C}(y, z), \mathrm{C}(x, y, z) \tag{1.6}$$

(we may ignore the order of strings in each pair and in the triple, since by varying the order of components in a tuple, we change the Kolmogorov complexity of the tuple by only an additive term $O(1)$). Besides these "canonical" quantities we have several values of the conditional complexities (e.g., $\mathrm{C}(x \mid y)$, $\mathrm{C}(x \mid y, z)$, $\mathrm{C}(x, y \mid z)$) and the mutual information (e.g., $I(x : y)$, $I(x : y, z)$). We also have the quantities of the conditional mutual information, e.g., $I(x : y \mid z)$. There is one more (a less standard) information quantity called *the mutual information of the triple*, denoted by $I(x : y : z)$. It has many equivalent (up to a logarithmic term) definitions; the most symmetric one is

$$\begin{aligned} I(x : y : z) \quad := \quad & \mathrm{C}(x) + \mathrm{C}(y) + \mathrm{C}(z) \\ & - \mathrm{C}(x, y) - \mathrm{C}(x, z) - \mathrm{C}(y, z) \\ & + \mathrm{C}(x, y, z). \end{aligned}$$

Using the Kolmogorov–Levin theorem, we can express each of these quantities as a linear combination of the primary values (1.6), e.g.,

$$\begin{aligned} I(x, y : z) \quad &=^{+} \quad \mathrm{C}(x, y) + \mathrm{C}(z) - \mathrm{C}(x, y, z), \\ I(x : y \mid z) \quad &=^{+} \quad C(x, z) + C(y, z) - \mathrm{C}(x, y, z) - \mathrm{C}(z), \end{aligned}$$

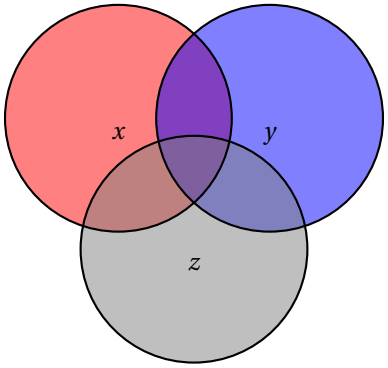

Figure 1.2: The diagram representing the information quantities of a triple of strings (x, y, z).

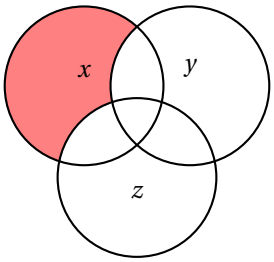

Figure 1.3: The area corresponding to $C(x \mid y, z)$.

and so on.

Venn diagrams help to deal with this variety of information quantities, see Fig. 1.2. In this picture the Kolmogorov complexities of x, y, and z correspond to the areas of the circles (red, blue, and gray respectively); the Kolmogorov complexity of a pair or a triple corresponds to the area occupied by the union of two or three circles.

The conditional complexities are represented by the differences of the corresponding areas: $C(x \mid y, z)$ is shown in Fig. 1.3, $C(x, y \mid z)$ in Fig. 1.4, and so on. The values of the mutual information are represented by the corresponding intersections. For example, the value of $I(x : y)$ is represented by the intersection of the red and the blue circles in Fig. 1.2; the value of $I(x, y : z)$ is shown in Fig. 1.5; the value of $I(x : y \mid z)$ is shown in Fig. 1.6. The mutual information of the triple $I(x : y : z)$ is shown in Fig. 1.7.

So far the geometric representation in Fig. 1.1 is only a simple way to visualize the information quantities, without any "added value cost." However, these diagrams simplify the manipulation with information quantities. For example, using these

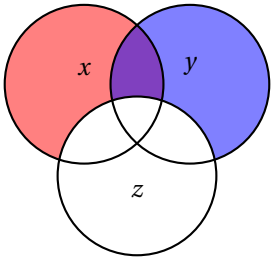

Figure 1.4: The area corresponding to $\mathrm{C}(x, y \mid z)$.

pictures it is easy to guess that

$$\begin{aligned} \mathrm{C}(x, y \mid z) &=^+ \mathrm{C}(x \mid y, z) + \mathrm{C}(y \mid x, z) + I(x : y \mid z), \\ I(x, y : z) &=^+ I(x : z \mid y) + I(y : z \mid x) + I(x : y : z), \end{aligned}$$

or

$$I(x : y : z) =^+ I(x : y) - I(x : y \mid z).$$

Once such an equation is seen from the diagram, its formal proof can be easily obtained as a combination of several instances of the basic equation from the Kolmogorov–Levin theorem.

Notice that all information quantities for a triple of string can be represented as linear combinations of seven "atomic" quantities

$$\begin{aligned} &\mathrm{C}(x \mid y, z),\ \mathrm{C}(y \mid x, z),\ \mathrm{C}(z \mid x, y), \\ &I(x : y \mid z),\ I(x : z \mid y),\ I(y : z \mid x), \\ &I(x : y : z) \end{aligned} \tag{1.7}$$

(one guy for each "atomic" area in Fig. 1.2). Thus, we have two useful "coordinate systems" that permit to express all standard information quantities for a triple of strings: the "canonical" quantities (1.6) and the "atomic" quantities (1.7). Obviously, all quantities (1.6) are non-negative. The quantities (1.7) are also non-negative (up to a logarithmic term) with one exception: the mutual information of the triple $I(x : y : z)$ can be far below zero. For example, for a pair of independent n-bit strings x, y of maximal complexity, and for z defined as the bitwise XOR of x and y, we have $I(x : y : z) \approx -n$.

For an arbitrary $n > 3$ we can take an n-tuple of strings $(x_1, \ldots, x_n)$ and consider information quantities of all kind involving these strings. And again, similarly to the case of $n = 2$ and $n = 3$, all usual information quantities (unconditional and conditional complexities and mutual information) can be represented as linear combinations of the "canonical" quantities

$$\mathrm{C}(x_{i_1}, \ldots, x_{i_k}) \tag{1.8}$$

(here we take all $2^n - 1$ non-empty subsets of indices $1 \le i_1 < \ldots < i_k \le n$). We can also define a general version of $2^n - 1$ "atomic" coordinates, similar to (1.7) (though drawing a visual diagram similar to Fig. 1.1 and Fig. 1.2 is not that simple for $n > 3$).

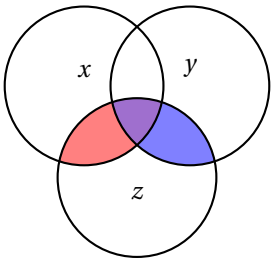

Figure 1.5: The area corresponding to $I(x, y : z)$.

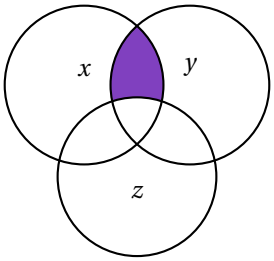

Figure 1.6: The area corresponding to $I(x : y \mid z)$.

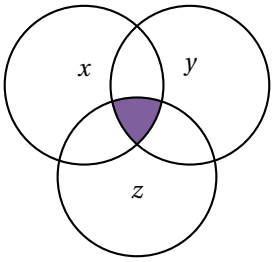

Figure 1.7: The area corresponding to $I(x : y : z)$.

1.4 Universal Information Inequalities

We mentioned in the previous section that for every n-tuple of strings $(x_1, \dots, x_n)$ we have $2^n - 1$ "canonical" information quantities (1.8). We refer to this vector of real numbers as *the complexity profile* of $(x_1, \dots, x_n)$. A fundamental problem of algorithmic information theory is to find a characterization of the set of all complexity profiles for every integer n. (Roughly speaking, which vectors can be realized as complexity profiles?) This problem is rather simple for $n = 2$ and $n = 3$ (see [j16]), but it remains widely open for $n > 3$. In what follows we discuss a dual version of this problem — the problem of linear information inequalities.

It is well-known that the following inequalities are true for all x, y, z:

$$\begin{aligned}
&\mathrm{C}(x) \le^+ \mathrm{C}(x, y),\\
&\mathrm{C}(x, y) \le^+ \mathrm{C}(x) + \mathrm{C}(y),\\
&\mathrm{C}(x, y, z) + \mathrm{C}(z) \le^+ \mathrm{C}(x, z) + \mathrm{C}(y, z).
\end{aligned}$$

The first of these inequalities is trivial: the optimal description of x is not longer than the optimal description of (x, y) (this inequality holds up to an additive constant, even without logarithmic terms). The second inequality claims that a pair of descriptions for x and y can be joint in a description for (x, y) (with a logarithmic overhead). The last inequality becomes more intuitive if we rewrite it (using the Kolmogorov–Levin theorem) to the form

$$\mathrm{C}(x, y \mid z) \le^+ \mathrm{C}(x \mid z) + \mathrm{C}(y \mid z),$$

and observe that a pair of programs translating z to x and z to y can be joined in one program translating z to $\langle x, y\rangle$ (again, with a logarithmic overhead).

These three inequalities rewrite to more compact forms

$$\begin{aligned}
&\mathrm{C}(y \mid x) \ge^+ 0,\\
&I(x : y) \ge^+ 0,\\
&I(x : y \mid z) \ge^+ 0,
\end{aligned}$$

respectively. These inequalities claim that every "atom" in Fig. 1.1 is non-negative.

Of course, these inequalities remain valid of we substitute any tuples of strings instead of individual strings x, y, z. Thus, for all $(x_1, \dots, x_n)$ we have

$$\begin{aligned}
&\mathrm{C}(x_V) \le^+ \mathrm{C}(x_W) \text{ for } V \subset W,\\
&\mathrm{C}(x_{V\cup W}) \le^+ \mathrm{C}(x_V) + \mathrm{C}(x_W), \text{ for all } V, W,\\
&\mathrm{C}(x_{V\cup W}) + \mathrm{C}(x_{V\cap W}) \le^+ \mathrm{C}(x_V) + \mathrm{C}(x_W) \text{ for all } V, W
\end{aligned} \tag{1.9}$$

(the properties of monotonicity, subadditivity, and submodularity). The inequalities of type (1.9) are usually called *basic information inequalities*.

For a triple of strings (x, y, z) the basic inequality claim that each "atom" in Fig. 1.8 except for the central area (corresponding to $I(x : y : z)$) is non-negative,

$$\begin{aligned}
&\mathrm{C}(x \mid y, z) \ge^+ 0,\ \mathrm{C}(y \mid x, z) \ge^+ 0,\ \mathrm{C}(z \mid x, y) \ge^+ 0,\\
&I(x : y \mid z) \ge^+ 0,\ I(x : z \mid y) \ge^+ 0,\ I(y : z \mid x) \ge^+ 0,
\end{aligned}$$

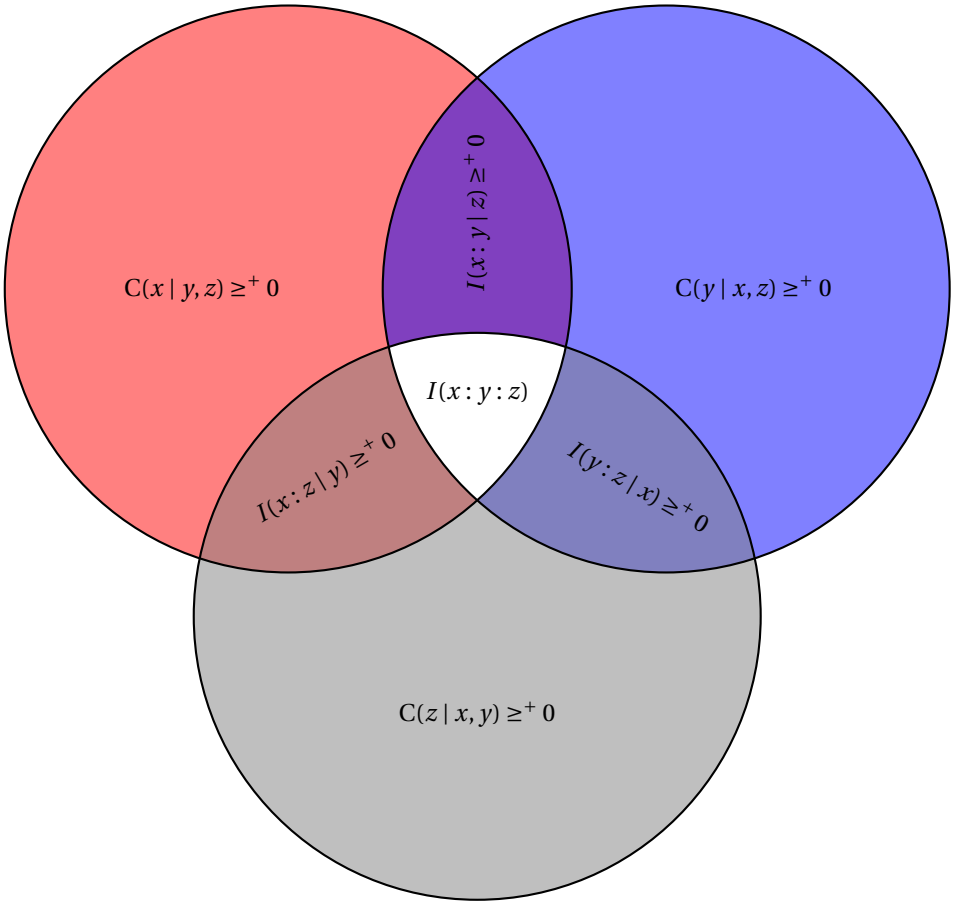

Figure 1.8: The atomic information quantities of a triple of strings (x, y, z): six of them are always non-negative. The quantity shown in white can be negative.

and also the sums corresponding to the pairwise mutual information are non-negative,

$$I(x : y \mid z) + I(x : y : z) \geq^+ 0,$$
$$I(x : z \mid y) + I(x : y : z) \geq^+ 0,$$
$$I(y : z \mid x) + I(x : y : z) \geq^+ 0.$$

(The other basic inequalities can be reduced to the nine inequalities listed above.)

Thus, the class of basic inequalities for 2-tuples of strings reduces to 3 independent inequalities, and the class of basic inequalities for 3-tuples of strings reduces to 9 independent inequalities (see [j16], [87] for detail).

Obviously, any linear combination of basic inequalities (with non-negative real coefficients) is again a universal information inequality, i.e., an inequality that is valid (up to a logarithmic term) for all tuples of strings. In what follows we discuss

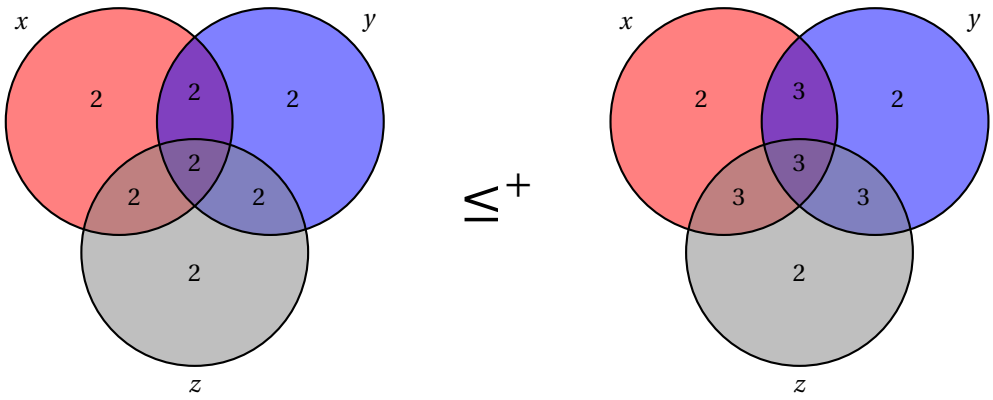

Figure 1.9: Diagrams representing inequality (1.10).

two simple (and pretty standard) examples.

Example 1. The inequality

$$2\mathrm{C}(x,y,z) \leq^+ \mathrm{C}(x,y) + \mathrm{C}(x,z) + \mathrm{C}(y,z) \tag{1.10}$$

is valid as a positive linear combination of basic inequalities. To see this, we represent the left-hand side and the right-hand side of this inequality in Venn-like diagrams, see Fig. 1.9. The difference between the left-hand side and the right-hand side of this inequality is shown in Fig. 1.10. In other words, we have reduced (1.10) to the inequality

$$I(x:y \mid z) + I(x:z \mid y) + I(y:z \mid x) + I(x:y:z) \geq^+ 0.$$

The sum of the four "atomic" quantities with non-zero coefficients shown in Fig. 1.10 can be represented as the sum of $I(x:y)$, $I(x:z \mid y)$, and $I(y:z \mid x)$. Each of these three quantities is non-negative (up to a logarithmic term), and therefore the left-hand side of (1.10) is indeed not greater than the right-hand side of this inequality.

This argument can be explained more formally, without diagrams. We start with three basic inequalities $I(x:y) \geq^+ 0$, $I(x:z \mid y) \geq^+ 0$, and $I(y:z \mid x) \geq^+ 0$. The sum of these inequalities rewrites to

$$\begin{array}{rl} & \mathrm{C}(x) + \mathrm{C}(y) - \mathrm{C}(x,y) \geq^+ 0 \\ + & \mathrm{C}(x,y) + \mathrm{C}(y,z) - \mathrm{C}(x,y,z) - \mathrm{C}(y) \geq^+ 0 \\ & \mathrm{C}(x,y) + \mathrm{C}(x,z) - \mathrm{C}(x,y,z) - \mathrm{C}(x) \geq^+ 0 \\ \hline & \mathrm{C}(x,y) + \mathrm{C}(x,z) + \mathrm{C}(y,z) - 2\mathrm{C}(x,y,z) \geq^+ 0, \end{array}$$

which is equivalent to (1.10).

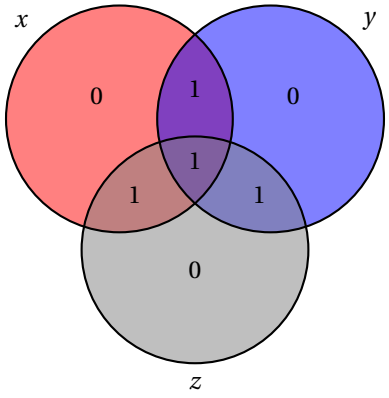

Figure 1.10: The difference between the right-hand side and the left-hand side of (1.10).

Example 2. Similarly we can prove the inequality

$$\mathrm{C}(z) \leq^+ \mathrm{C}(z \mid x) + \mathrm{C}(z \mid y) + I(x : y). \tag{1.11}$$

The left-hand side and the right-hand side of this inequality are represented in the diagrams in Fig. 1.11. The difference between the left-hand side and the right-hand side of this inequality is shown in Fig. 1.12. The two o "atomic" quantities with non-zero coefficients in Fig. 1.12 are $I(x : y \mid z)$ and $\mathrm{C}(z \mid x, y)$. Both of them are non-negative due to the basic information inequalities.

Let us rephrase the same argument more formally. The basic inequalities $I(x : y \mid z) \geq^+ 0$ and $\mathrm{C}(z \mid x, y) \geq^+ 0$ rewrite to

$$\begin{aligned} &\mathrm{C}(x,z) + \mathrm{C}(y,z) - \mathrm{C}(x,y,z) - \mathrm{C}(z) \geq^+ 0, \\ &\mathrm{C}(x,y,z) - \mathrm{C}(x,y) \geq 0. \end{aligned}$$

The sum of these inequalities is

$$\mathrm{C}(x,z) + \mathrm{C}(y,z) - \mathrm{C}(x,y) - \mathrm{C}(z) \geq^+ 0,$$

which easily rewrites to (1.11).

The two examples above show that the proof of corollaries of basic information inequalities is a simple mechanical procedure. If more than three variables are involved, we may need to use a linear programming solver instead of a Venn-type diagram, but the general principle remains simple. In the subsequent text, we skip detailed proofs of inequalities that can be reduced to combinations of basic information inequalities.

In general, we say that an inequality with coefficients λ_V is a *universal inequality for the Kolmogorov complexity*, if

$$\sum_{V \subset \{1,\dots,k\}} \lambda_V \mathrm{C}(x_V) \geq^+ 0 \tag{1.12}$$

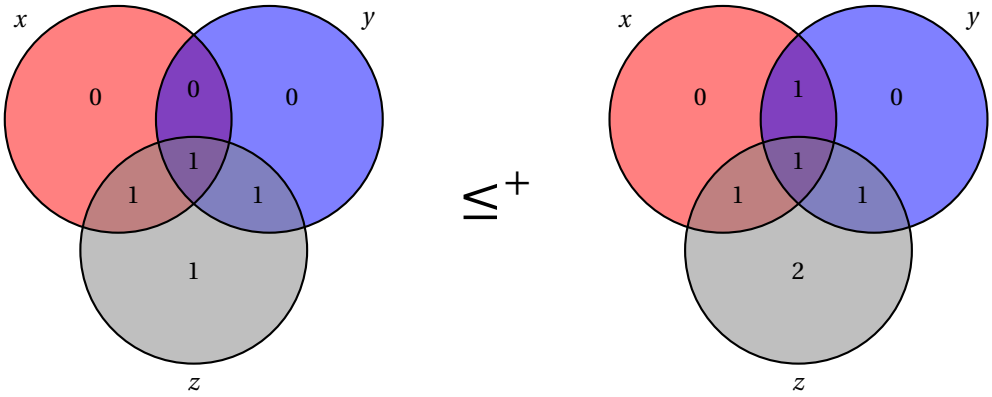

Figure 1.11: Diagrams representing inequality (1.11).

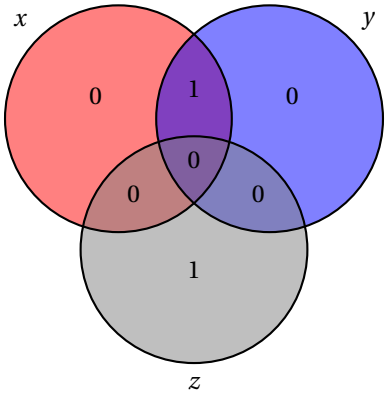

Figure 1.12: The difference between the right-hand side and the left-hand side of (1.11).

for all $x_1, \dots, x_k$. (In the implicit $O(\log n)$ term in (1.12) we assume $n = \mathrm{C}(x_1, \dots, x_k)$, and the multiplicative factor hidden in the $O(\cdot)$ notation does not depend on the strings involved.)

Major Open Problem: Characterize the universal inequality for the Kolmogorov complexity (1.12).

Only a partial solution of this problem is known:

Theorem 4 ([j16]). *For $n \le 3$ the class of all universal inequality for the Kolmogorov complexity of n-tuples of strings coincides with the class of all non-negative linear combinations of the basic information inequalities* (1.9).

Theorem 4 cannot be extended to $n > 3$, see inequality (1.14) below. In other words, it is known that for every $n > 3$ there exist universal information inequalities that cannot be represented as a combination of basic inequalities, though a complete description of these inequalities remains elusive.

The *Major Open Problem* above can be naturally translated in the language of Shannon's theory. We say that an inequality with coefficients λ_V is a *universal inequality for Shannon's entropy*, if

$$\sum_{V \subset \{1,\dots,n\}} \lambda_V H(X_V) \ge^+ 0 \tag{1.13}$$

for all joint distributions $(X_1, \dots, X_n)$ with a finite range. We can ask a similar question on a characterization of the universal inequalities for Shannon's entropy. It turns out that these two questions are equivalent, i.e., the same linear inequalities are valid for Shannon's entropy and for Kolmogorov complexity:

Theorem 5 ([j16]). *For every integer $n \ge 0$ and for every set of $2^n - 1$ coefficients λ_V ($V \subset \{1, \dots, n\}$), inequality* (1.12) *is valid for all binary strings $x_1, \dots, x_n$ if and only if inequality* (1.13) *is valid for all jointly distributed $(X_1, \dots, X_n)$.*

Thus, though for $n > 3$ we do not know what inequalities are valid for the Kolmogorov complexities of n-tuples of strings and for Shannon's entropies of n-tuples of random variables, we know that these classes of inequalities are the same.

In 1998 Z. Zhang and R.W. Yeung came up with the first example of a universal inequality for Shannon's entropy that cannot be represented as a combination of basic inequalities, [44]. Technically, this inequality involves four random variables and can be expressed in standard notation as follows:

$$\begin{aligned} I(X:Y) \le 2I(X:Y \mid A) + I(X:Y \mid B) + I(A:B) \\ + I(X:A \mid Y) + I(Y:A \mid X) \end{aligned} \tag{1.14}$$

A more symmetric version (with 5 variables) of (1.14) was later proven in [j13]:

$$\begin{aligned} I(X:Y) \le I(X:Y \mid A) + I(X:Y \mid B) + I(A:B) \\ + I(X:Y \mid Z) + I(X:Z \mid Y) + I(Y:Z \mid X) \end{aligned} \tag{1.15}$$

(inequality (1.14) is an instantiation of (1.15) for $A = Z$).

Given Theorem 5, we can conclude that the counterparts of (1.14) and (1.15) are universal inequalities for the Kolmogorov complexity (that are valid, as usual, up to a logarithmic additive term). The universal inequalities for Shannon's entropy and for Kolmogorov complexity that cannot be represented as a convex combination of basic inequalities are called in the literature *non-Shannon-type linear information inequalities*. Since 1998 many examples of non-Shannon-type information inequalities were discovered, see [84]. Most of these inequalities were originally proven for Shannon's entropy. Due to Theorem 5, each of these inequalities can be reformulated as an inequality for Kolmogorov complexity.

1.5 Stochastic Strings and Tuples of Strings

Though the notion of (α,β)-stochasticity is not a central subject of this manuscript, we extensively use it as a powerful technical tool. In this section, we recall the definition and the basic properties of stochastic objects.

The idea of a stochastic object was proposed by Kolmogorov in the 1970s (see [19]). The following definition first appeared in print in [21].

Definition 4. *A string x is called* (α,β)-stochastic *if there is a finite set S containing x such that* $\mathrm{C}(S)\le\alpha$ *and* $\mathrm{C}(x\mid S)\ge\log(\#S)-\beta$.

This definition is usually applied with rather small α and β (say, with α and β logarithmic in complexity of x). The intuitive interpretation of this definition is the idea of "explanation" of a random object x. Here S is a "simply explainable" set, and x is a "typical" element in this set. Thus, if we get x as an observation or a measurement of a "natural phenomenon" (in one or another sense), then it is plausible to conjecture that the global natural phenomenon is the set S, and we observed just a typical instance of this phenomenon.

The property $\mathrm{C}(S)\le\alpha$ means that there is a program of size at most α that prints the list of all elements of S and stops. A distinguishable completion is crucial, it is not enough to have an algorithm that enumerates all elements of S but never shows manifestly the end of the process.

For example, if x is a binary string of length n and $\mathrm{C}(x)\ge n-O(\log n)$, then x is $(O(\log n),O(\log n))$-stochastic since we can let S be the set of all strings of length n.

Definition 4 applies to tuples of strings as well as to individual strings. In fact, in most applications of Kolmogorov complexity, we deal with various $(O(\log n),O(\log n))$-stochastic objects (i.e., with typically behaving constructive objects chosen from a set S that has a simple description), see many instances of the incompressibility argument in [76].

The very fact that some strings are not stochastic (for reasonably small α and β) is not trivial. It is known (see [21]) that if

$$2\alpha+\beta<n-O(\log n),$$

then there exists a string of length n that is not (α,β)-stochastic. An example of a highly non-stochastic object is the n-bits prefix of Chaitin's Omega number, [18] (see also the survey [100]).

For every string x, there exists a similar stochastic "sibling" x'. More precisely, we say that x and x' are δ-equivalent, if

$$\mathrm{C}(x \mid x') < \delta \text{ and } \mathrm{C}(x' \mid x) \le \delta.$$

We observe that for every string x of complexity n, there exists an $(O(\log n), O(\log n))$-stochastic x' such that x and x' are $O(\log n)$-equivalent. Indeed, for any given x we can define its stochastic sibling x' as the shortest description (in the sense of the optimal description method U, see p. 27) of x.

The same time, a tuple combined of stochastic objects can be highly non-stochastic as a whole. Moreover, we generally cannot convert a non-stochastic tuple in a stochastic one by changing each component of the tuple to an $O(\log n)$-equivalent sibling:

Theorem 6 ([j9]). *Let $\alpha, \beta, \gamma, C > 0$ be real numbers such that $\alpha + \beta > \gamma$ and $\alpha, \beta < \gamma$. Then for all large enough n there exist strings x_1, x_2 such that*

- $\mathrm{C}(x_1) =^+ \alpha n$,
- $\mathrm{C}(x_2) =^+ \beta n$,
- $\mathrm{C}(x_1, x_2) =^+ \gamma n$,

and there is no $(C\log n, C\log n)$-stochastic pair (x_1', x_2') where x_i is $(C\log n)$-equivalent to x_i' for $i = 1, 2$.

In Section 1.8 we will see that stochastic tuples of strings enjoy nice properties of extracting the mutual information. Theorem 6 explains why this result cannot be easily extended to non-stochastic tuples. On the other hand, by restricting ourselves to the class of stochastic objects, we do not change the class of valid information inequalities. More specifically, we say that an inequality with coefficients λ_V (indexed by nonempty subsets $V \subset \{1, \dots, k\}$) holds for all $(O(\log n), O(\log n))$-stochastic tuples, if for every real number C there exists a D such that for every $(C\log n, C\log n)$-stochastic k-tuple $\langle x_1, \dots, x_k \rangle$

$$\sum_{V \subset \{1,\dots,k\}} \lambda_V \mathrm{C}(x_V) + D\log \mathrm{C}(x_1, \dots, x_k) \ge 0. \tag{1.16}$$

Theorem 7. *For every integer $k \ge 0$, for every set of $2^k - 1$ coefficients λ_V ($V \subset \{1, \dots, k\}$), inequality* (1.12) *is valid for all binary strings $x_1, \dots, x_n$ if and only if the inequality with the same coefficients is valid for $(O(\log n), O(\log n))$-stochastic tuples.*

This theorem was not proven explicitly in [j16], though the argument from [j16, theorem 1] easily adapts to this setting, as we show below.

Proof: From Theorem 5 we know that the same classes of linear inequalities are valid for Kolmogorov complexity and for Shannon's entropy. Also all universal inequalities for Kolmogorov complexity hold for all $(O(\log n), O(\log n))$-stochastic tuples. It remains to show that the inequalities for Kolmogorov complexity that are valid for all $(O(\log n), O(\log n))$-stochastic tuples also apply to Shannon's entropy.

Let $(X_1, \ldots, X_k)$ be jointly distributed random variables. We are going to show that the counterparts of inequality (1.16) that are valid for the stochastic tuples of strings hold true for Shannon's entropies involving $(X_1, \ldots, X_k)$.

We assume that each X_i is distributed on a finite alphabet Σ_i. We fix an integer N and consider $(k \times N)$-matrices

$$\begin{pmatrix} a_{11} & a_{12} & \ldots & a_{1N} \\ a_{21} & a_{22} & \ldots & a_{2N} \\ \ldots\ldots\ldots\ldots\ldots\ldots \\ a_{k1} & a_{k2} & \ldots & a_{kN} \end{pmatrix}$$

where the elements of each ith row are letters from Σ_i, $i = 1, \ldots, k$. We restrict ourselves to the matrices where each column

$$\begin{pmatrix} \sigma_1 \\ \sigma_2 \\ \vdots \\ \sigma_k \end{pmatrix}$$

appears in the matrix

$$\mathrm{prob}[X_1 = \sigma_1 \,\&\, \ldots \,\&\, X_k = \sigma_k] \cdot N + O(1)$$

times (the $O(1)$ term is added to handle the rounding and guarantee that this set of matrices is not empty). We define S_N as the set of all k-tuples composed of rows of these matrices.

It can be verified (see [j16] or [j17] for details) that the cardinality of S_N is equal to $2^{H(X_1,\ldots,X_k)N+O(\log n)}$ and for all tuples $\langle x_1, \ldots, x_k \rangle$ in S_N we have

$$\mathrm{C}(x_V) \leq^+ N \cdot H(X_V) \text{ for every set of indices } V \subset \{1, \ldots, k\};$$

moreover, for most tuples in S_N this bound is tight, i.e.,

$$\mathrm{C}(x_V) =^+ N \cdot H(X_V) \quad \text{for every set of indices } V \subset \{1, \ldots, k\}. \tag{1.17}$$

Now we fix an arbitrary tuple $\langle x_1, \ldots, x_k \rangle$ in S_N that satisfies (1.17). This tuple is $(O(\log N), O(\log N))$-stochastic (as a typical element of a "simple" set S_N). Hence, it satisfies every inequality (1.16) valid for stochastic tuples. Combining (1.16) with (1.17) we obtain:

$$\sum_{V \subset \{1,\ldots,k\}} \lambda_V H(x_V) \cdot N + O(\log N) \geq 0.$$

Dividing this inequality by N and letting N go to infinity we get

$$\sum_{V \subset \{1,\ldots,k\}} \lambda_V H(x_V) \geq 0,$$

i.e., the inequality with coefficients λ_V holds for the entropies of the initial distribution. □

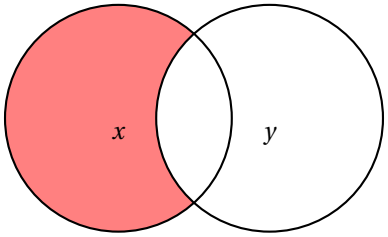

Figure 1.13: The diagram representing the information quantities of a pair (x, y).

1.6 The Conditional Description Theorems: Absolute and Resource-Bounded Versions

In this section, we discuss in more detail the information quantity $C(x \mid y)$, which is shown in the Venn diagram in Fig. 1.13. By definition, this value is the length of the shortest program (in the optimal programming language) that translates y to x. However, a naive interpretation of the diagram suggests that this quantity may have a more explicit interpretation. We would like to think of $C(x \mid y)$ as a part of x that lays outside y. It turns out that this naive idea can be implemented in a formal statement. More precisely, we can find a "fingerprint" of x of size close to $C(x \mid y)$ such that x can be reconstructed given this fingerprint and y. This statement is referred to as Muchnik's theorem on conditional descriptions:

Theorem 8 (An. Muchnik, [55])**.** *Let x and y be two binary strings, $\mathrm{C}(x) \le n$ and $C(x \mid y) \le k$. Then there exists a string p such that*

- $\mathrm{C}(x \mid p, y) = O(\log n)$,
- $\mathrm{C}(p) \le k + O(\log n)$,
- $\mathrm{C}(p \mid x) = O(\log n)$.

The constants hidden in $O(\cdot)$ notations do not depend on x, y and k, n.

Remark 1. The inequality $\mathrm{C}(p) \le k + O(\log n)$ in the statement of the theorem can be replaced by the condition $|p| \le k$.

Remark 2. Without the last condition, this theorem would be a trivial rewording of the definition of conditional Kolmogorov complexity. The whole statement is interesting only because we require that $\mathrm{C}(p \mid x) \approx 0$.

This theorem is an algorithmic counterpart of the asymmetric Slepian–Wolf theorem (which is a classic result in Shannon's information theory, [13]). Muchnik's theorem on conditional descriptions can be naturally interpreted in terms of multi-source information theory. Assume that some person Bob knows y and wants to know x. Another person Alice knows x and wants to send some message p to Bob so that the latter could reconstruct x. How long should be this message? Does Alice

need to know y to be able to provide such a message? Muchnik's theorem suggests a kind of answer to these questions. Indeed, the complexity of a piece of information p that together with y allows Bob to reconstruct x must be at least $C(x \mid y)$. It is easy to see that the minimal p can be found (with logarithmic advice) when given both strings x and y. Muchnik's theorem claims more: it explains that in fact Alice does not need to know y, she can provide p given only x (and, again, some logarithmic advice).

Remark 3. Bennett *et al.* suggested in [42] a statement similar to Muchnik's theorem above: they showed that an almost shortest program p translating y to x can be obtained as a function of x; however, to compute this function we need an oracle of size $O(n)$ (this oracle depends on n and k but not on specific x and y). Thus, in Theorem 8 we need $O(\log n)$ bits of advice to obtain p from x, and this advice depends on y, while in [42, Theorem 3.11] we need $O(n)$ bits of advice, though this advice depends only on k and n.

Theorem 8 has an interesting generalization to the case of many conditions.

Theorem 9 ([55])**.** *Let x and $y_1, \dots, y_m$ be binary strings, $C(x) \le n$ and $C(x \mid y_i) \le k_i$ for $i = 1, \dots, m$. Then there exists a string p such that*

- $C(x \mid p_{[1:k_i]}, y_i) = O(\log n + \log m)$ *for* $i = 1, \dots, m$,
- $C(p) \le \max\limits_{i=1,\dots,m} \{k_i\} + O(\log n + \log m)$,
- $C(p \mid x) = O(\log n + \log m)$.

Here $p_{[1:k_i]}$ denotes the prefix of p of length k_i. Since the $O(\cdot)$ terms in the inequalities above involve $\log m$, this statement is usually applied with $m = \mathrm{poly}(n)$.

Speaking informally, Theorem 9 claims that one fingerprint p obtained from x is enough to reconstruct x, given any of the conditions y_i. Moreover, to retrieve x from y_i, we need only the first k_i bits of this fingerprint.

After the publication of [55], other authors suggested several alternative proofs of Theorem 8. Some versions of these proofs work with slightly different conditions (e.g., with resource-bounded versions of Kolmogorov complexity). All known proofs of Muchnik's theorem use more or less explicitly the following scheme. We construct some family of functions

$$f_i : \{0,1\}^n \to \{0,1\}^k$$

such that for all sets $S \subset \{0,1\}^n$ of size $\approx 2^k$ the functions f_i are "almost one-to-one" correspondences between S and $\{0,1\}^k$. More precisely, for most $x \in S$ at least one of f_i-images determines x almost-uniquely (there are very few other $x' \in S$ such that $f_i(x) = f_i(x')$). Then we can take

$$S = \{x' \,:\, C(x' \mid y) \le k\}$$

and choose a suitable values $f_i(x)$ as the value of p required in Theorem 8. The difficult part of the proof is to construct the family of functions f_i appropriate for all S.

The general scheme sketched above can be explained in various combinatorial terms and implemented with various techniques. For instance, [j8] reduces the statement of Theorem 8 to a combinatorial problem called *on-line matching*. This problem is an on-line version of the classical matching problem for bipartite graphs. Muchnik's theorem can be rephrased as some combinatorial statement about on-line matchings, and then a proof of this combinatorial statement implies Muchnik's theorem (see [j8, Section 2]). Another approach to Muchnik's theorem is based on the notion of extractors. In what follows we discuss the technique of extractors and its variations in some detail.

First of all we recall the conventional definition of an extractor graph. Let G be a bipartite graph with N vertices in the left part and M vertices in the right part. We allow parallel edges. We assume that all vertices of the left part have the same degree D. We fix an integer $K > 0$ and a real number $\varepsilon > 0$ and define an extractor as follows.

Definition 5. *A bipartite graph G is an extractor with parameters $(N, M, D, K, \varepsilon)$ if for all subsets S of its left part such that $\#S \geq K$ and for all subsets Y of the right part the inequality*

$$\left| \frac{\#E(S,Y)}{D \cdot \#S} - \frac{\#Y}{M} \right| < \varepsilon \tag{1.18}$$

holds, where $E(S,Y)$ stands for the set of edges between S and Y.

This definition may be rephrased as follows: We consider the uniform distribution on a set S of left-part vertices. We take a random vertex x in S and a random neighbor y of x. Then the probability to end up in Y is equal to $\#E(S,Y)/(D \cdot \#S)$. And we claim that this probability is ε-close to $\#Y/M$, which is exactly the probability to end up in Y by taking a random vertex in the right part.

An extractor can be understood as a function of two arguments Ext that takes as an input an index $x = 1, \ldots, N$ (a vertex of the left part) and $j = 1, \ldots, D$ (an index of the neighbor), and returns a $y = 1, \ldots, M$ (a vertex of the right part) that is the jth neighbor of x. If we have at our disposal a "poor" source of randomness, which is a uniform distribution on a K-element subset of the left part of the graph, and a perfectly uniform distribution on the set $1, \ldots, D$, then by applying Ext to this pair we get a distribution $Ext(x, j)$ that is ε-close to the "perfectly random" uniform distribution on the right part of the graph. In other words, we "extract" the poor randomness from the given distribution by adding a moderate number of fresh perfectly random bits.

The definition of extractors is interesting when $M \approx K$ (we extract almost all randomness from the "poorly random" distribution) and $D \ll N$ (we need only very few perfectly random bits). A probabilistic argument shows that extractors with some nice parameters do exist:

Theorem 10 (see [48])**.** *For all K, N, M and ε such that $1 < K \leq N$, $M > 0$, $\varepsilon > 0$, there exists an $(N, M, D, K, \varepsilon)$-extractor with*

$$D = \max\left\{ O\left(\frac{M}{\varepsilon^2 K}\right),\ O\left(\frac{1 + \log \frac{N}{K}}{\varepsilon^2}\right) \right\}.$$

Moreover, this bound for D is optimal up to factor of $(1/\varepsilon)^{O(1)}$.

Most applications in computer science require an *explicit* construction of extractors. By an explicit construction we mean that for arbitrary values of n and k (where $k \le n$) there exists an extractor with parameters $N = 2^n$ and $K = 2^k$, the corresponding values of parameters M and ε can be computed in time poly(n), and the extractor as a function of two arguments is also computable in time poly(n). By now, no explicit constructions of extractors with optimal parameters have been invented. In every known explicit construction the trade-off between parameters is worse than in Theorem 10. In the sequel, we use the following theorem:

Theorem 11 ([67])**.** *For all k, n and ε such that $1 < k \le n$ and $\varepsilon > 1/\mathrm{poly}(n)$, there exists an explicit $(N, M, D, K, \varepsilon)$-extractor with parameters $N = 2^n$, $M = D \cdot 2^k$ and $D = 2^{O((\log n \log\log n)^2)}$.*

For the sake of brevity, in the sequel, we use a slightly weaker bound $O(\log^3 n)$ instead of $O((\log n \log\log n)^2)$.

It turns out that the definition of extractors perfectly matches Muchnik's idea of conditional descriptions. We can prove Theorem 8 by taking an extractor Ext from Theorem 10 with suitably chosen parameter and by letting $p = Ext(x, j)$ for some value of j. This idea appeared in [49] (though without an explicit statement of Theorem 10). Theorem 9 can be proven similarly with a suitably defined *prefix extractors*, see [j8, Section 3.2–3.3] for details.

We can combine in the proof of Muchnik's theorem with an explicit construction of extractors from Theorem 11. This plan results in several versions of Theorem 8, with one or another resource-bounded variant of Kolmogorov complexity. We start with a version of of Muchnik's theorem for the space-bounded variant of Kolmogorov complexity:

Theorem 12 ([j8])**.** *Let x and y be binary strings and n, k and s be numbers such that $\mathrm{C}^{\infty,s}(x) < n$ and $\mathrm{C}^{\infty,s}(x \mid y) < k$. Then there exists a binary string p such that*

- $\mathrm{C}^{\infty,O(s)+\mathrm{poly}(n)}(x \mid p, y) = O(\log^3 n)$;
- $\mathrm{C}^{\infty,O(s)}(p) \le k + O(\log n)$;
- $\mathrm{C}^{\infty,\mathrm{poly}(n)}(p \mid x) = O(\log^3 n)$,

where all constants in O- and poly*-notation depend only on the choice of the optimal description method.*

The $O(\log^3 n)$ terms in the last theorem come from Theorem 11. The poly-logarithmic bound does not look natural here ; it seems to be an artifact of the proof. As soon as explicit extractors with optimal parameters are constructed, the proof of Theorem 12 from [j8] will imply a similar result with $O(\log n)$ terms.

Interestingly, in some cases, we can achieve virtually optimal bounds even without optimal explicit extractors. In [93] D. Musatov showed that for the most interesting type of the space bound $s = \mathrm{poly}(n)$, the terms $O(\log^3 n)$ in Theorem 12 can be reduced to $O(\log n)$. The proof suggested by Musatov employs a nice construction of pseudo-random functions based on the Nisan–Wigderson generator, [32]. In the next section, we revisit this construction in a different context.

Later, B. Bauwens, A. Makhlin, N. Vereshchagin, and M. Zimand suggested a version of Muchnik's theory with a polynomial time bound for $C(p \mid x)$:

Theorem 13 ([104]). *Let x and y be two binary strings, $C(x) \le n$ and $C(x \mid y) \le k$. Then there exists a string p such that*

- $C(x \mid p, y) = O(\log n)$,
- $C(p) \le k + O(\log n)$,
- $C^{\mathrm{poly}(n),\infty}(p \mid x) = O(\log n)$.

The constants hidden in $O(\cdot)$ notations do not depend on x, y and k, n.

Remark 4. A subtle argument in [104] shows that even with a non-optimal explicit extractor from Theorem 11 we can get for $C^{\mathrm{poly}(n),\infty}(p \mid x)$ the bound $O(\log n)$ (instead of a weaker threshold $\mathrm{polylog}(n)$, which might be expected).

Remark 5. The statement of Theorem 13 is in some sense non-symmetric: it involves a time-bounded version of Kolmogorov complexity only for the term $C(p \mid x)$ but not for the other complexity terms.

One more variant of Muchnik's theorem is due to M. Zimand:

Theorem 14 ([102], [103]). *There exists a probabilistic algorithm that on input x, k, ϵ, where x is a string, k is a positive integer and $\epsilon > 0$, returns a string p of length $k + O(\log(n/\epsilon))$, and for every y, if $k \ge C(x \mid y)$, then with probability $(1-\epsilon)$, p is a program for x given y. The algorithm is using $O(\log(n/\epsilon))$ random bits, where n is the length of x.*

Moreover, the implied probabilistic algorithm can be made efficient (polynomial time) if we increase the overhead in the size of p from $O(\log(n/\epsilon))$ to $\mathrm{poly}(\log(n/\epsilon))$.

(A slightly weaker version appeared earlier in [91].) Loosely speaking, Theorem 14 claims that not only *one* fingerprint p obtained from x but *most of them* are suitable to reconstruct x given y as a condition. Theorem 14 involves no help bits: we do not need $O(\log n)$ bits of advice to reconstruct x given p and y. This theorem is used in Section 1.14.

In conclusion, we mention another version of Muchnik's theorem proven for a somewhat uncommon variant of Kolmogorov complexity based on poly-time Arthur–Merlin games (introduced in [61]). This proof extensively employs special properties of the extractor constructed by L. Trevisan in [47]. For details, we refer the reader to [j8, Section 3.5]

1.7 Digression: Bit-Probe Schemes

In this section, we discuss several results obtained with the technique from the paper by D. Musatov [93] mentioned in the previous section (a proof of a version of Muchnik's theorem for space-bounded Kolmogorov complexity). These results do

not involve Kolmogorov complexity explicitly — they deal with a sort of data structures[1]. More specifically, we construct a querying scheme for the static version of the *membership problem*. The aim is to represent a set $A \subset \{1,\dots,m\}$ by some data structure so that queries "$x \in A$?" can be replied fast. We focus on the case when the number of elements in the set $n = \#A$ is much less than the size m of the universe (e.g., $n = \exp\{\mathrm{poly}(\log\log m)\}$ or $n = m^{0.01}$).

In practical application, there are several popular data structures that represent sets: arrays, different types of height-balanced trees, hash tables, and so on. Arguably the simplest way to store a set is to keep in memory an array of m bits: the x-th bit in the data storage is equal to 1 if and only if $x \in A$. Then, to answer a query "$x \in A$?" we need to read from memory a single bit. The disadvantage of this simple approach is the size of the data structure: we keep in memory m bits, though there exist only $\binom{m}{n} = 2^{\Theta(n\log m)}$ different subsets A of size n in the m-elements universe.

Nowadays the standard practical solution for the membership problem is a more complex data structure proposed by Fredman, Komlós, and Szemerédi [22]. This scheme uses perfect hashing. A set is represented as a table of $O(n)$ words (hash-values) of size $\log m$ bits, and a query "$x \in A$?" requires to read $O(1)$ words from the memory. The space complexity of this construction is quite close to the lower bound $\Omega(\log\binom{m}{n})$. Further improvements of this scheme were suggested in [26], [45], [51]. Similar asymptotics of space complexity was achieved in *dynamic* data structures, which support fast update of the set stored in the database (see, e.g., the analysis of the cuckoo hashing scheme in [57], [60]). A subtle analysis of the space and bit-probe complexity for the membership problems was also given in [52] (in particular, [52] suggested a membership scheme based on bounded concentrator graphs). All these schemes require to read from the memory $O(\log m)$ bits to answer each query.

Another popular practical solution is Bloom's filter, [7]. This data structure requires only $O(n)$ bits, whatever is the size of the universe; to answer a query we need to read $O(1)$ bits from the memory. The drawback of Bloom's filter is that we get false answers to some queries. Only false positives answers are possible (for some $x \notin A$ Bloom's filter answers "yes"), but false negatives are not. When this technique is used in practice, it is usually believed that for a "typical" set A the fraction of false answers should be small. However, in many applications, we cannot fix *a priori* any natural probability distribution on the family of all possible sets A and on the space of possible queries.

An interesting alternative approach was suggested by Harry Buhrman, Peter Bro Miltersen, Jaikumar Radhakrishnan, and Venkatesh Srinivasan, [54]. They introduced randomness into the algorithm processing the queries. In their approach, the data structure is defined deterministically for each set A, but when a query is handled, we toss coins and read a randomly chosen bit from memory. In this model, we may return a wrong answer with a small probability. However, there is a critical difference with the Bloom's filter: in this scheme, we must correctly reply to the query "$x \in A$?" with probability close to 1 for *every* x (not for *most* x).

Buhrman, Miltersen, Radhakrishnan, and Venkatesh investigated schemes with two-sided and one-sided errors. In what follows, we focus mostly on one-sided

[1] This section can be skipped without loss of continuity.

errors: if $x \in A$, then the answer must always be correct, and if $x \notin A$, then a small probability of error is allowed. In this setting the trivial information-theoretic bound still applies: the size of the structure representing a set A cannot be less that $\log\binom{m}{n} = \Omega(n \log m)$ bits. Quite surprisingly, this bound can be achieved if we allow two-sided errors and use only single bit probe for each query, [54]. We refer to the scheme as the BMRV-scheme.

Theorem 15 (two-sided BMRV-scheme, [54]). *For any $\varepsilon > 0$ there is a scheme for storing subsets A of size at most n of a universe of size m using $O(\frac{n}{\varepsilon^2} \log m)$ bits so that any membership query "Is $x \in A$?" can be answered with error probability less than ε by a randomized algorithm which probes the memory at just one location determined by its coin tosses and the query element x.*

The size of the memory achieved in this theorem is only by a constant factor greater than the best possible. In fact, the trivial lower bound $\log\binom{m}{n}$ can be improved. For smaller probability of error, we need more memory.

Theorem 16 (lower bound from [54]). (*a*) *For any $\varepsilon > 0$ and $\frac{n}{\varepsilon} < m^{1/3}$, any randomized scheme with error ε that answers queries using one bitprobe must use space $\Omega(\frac{n}{\varepsilon \log 1/\varepsilon} \log m)$.*

(*b*) *Any scheme with* one-sided *error ε that answers queries using at most one bitprobe must use $\Omega(\frac{n^2}{\varepsilon^2 \log(n/\varepsilon)} \log m)$ bits of storage.*

Note that for one-sided error schemes the lower bound in this theorem is much stronger. Part (b) of the theorem implies that we cannot achieve the size of space $O(n \log m)$ with a one probe scheme and one-sided error. However we can get pretty close to this lower bound if we allow $O(1)$ probes instead of a single probe:

Theorem 17 (one-sided BMRV-scheme, [54]). *Fix any $\delta > 0$. There exists a constant t such that the following holds: There is a one-sided $\frac{1}{3}$-error randomized scheme that uses space $O(n^{1+\delta} \log m)$ and answers membership queries with at most t probes.*

The constructions in [54] are not explicit: given the list of elements A, the corresponding scheme is constructed (with some brute force search) in time $2^{\text{poly}(m)}$. Moreover, each membership query requires exponential in m computations.

The crucial part of the constructions in Theorem 15 is an unbalanced expander graph. The existence of a graph with required parameters was proven in [54] probabilistically. We know that such a graph exists and we can find it by brute force search, but we do not know how to construct it explicitly. If we have an effective construction of an expander with suitable parameters, we will get a more practical variant of the BMRV-scheme.

Since Bassalygo and Pinsker defined expanders [10], [12], many explicit (and poly-time computable) constructions of expander graphs were discovered, see an excellent survey [66]. However, most of the known explicit constructions are based on the spectral technique that is not suitable to get an expander of degree d with an expansion parameter greater than $d/2$, see [34]. Such an expansion rate is not good enough for the construction used in the proof of Theorem 15. There are only very few effective constructions of unbalanced graph with larger expansion parameter,

especially for highly unbalanced graphs. Some explicit version of the BMRV-scheme was suggested in [56] (this construction involves Trevisan's extractor, which can be used to build a good unbalanced expander, see [53]). The best known explicit construction of a highly unbalanced expander graph was presented in [79]. It is based on the Parvaresh–Vardy code with an efficient list decoding. Due to the very special structure if this expander, it enjoys nice property of effective decoding. Using this technique, the following variant of Theorem 15 was proven:

Theorem 18 ([79]). *For any $\delta > 0$ there exists a scheme for storing subset A of size at most n of a universe of size m using $n^{1+\delta} \cdot \mathrm{poly}(\log m)$ bits so that any membership query can be answered with error probability less than ε by a randomized algorithm which probes the memory at one location determined by its coin tosses and the query element x.*

Given the list of elements A, the corresponding storing scheme can be constructed in time $\mathrm{poly}(\log m, n)$. *When the storing scheme is constructed, a query for an element x can be calculated in time* $\mathrm{poly}(\log m)$.

In Theorems 15, 17, 18, a set A is encoded into a bit string, and when we process a query "$x \in A$?" we read from this string one bit (or $O(1)$ bits in Theorem 17) chosen by a randomized algorithm. Then we use the retrieved bit to decide whether x is an element of the set. Notice that in these computations we implicitly use more information than just a single bit extracted from the memory. Indeed, to make a query and to process the retrieved bit, we need to know the parameters of the scheme: the size n of the set A, the size m of the universe, and the allowed error probability ε. This auxiliary information is very short (it takes only $\log(m/\varepsilon)$ bits), and it does not depend on the stored set A. We assume that this information is somehow hardwired into the bitprobe scheme, or, so to speak, "cached" in advance by the algorithms processing queries.

We propose to consider a slightly looser setting, where some small information "cached" by the scheme can depend not only on n, m, and ε but also on the set A. Technically, the data stored in our scheme consists of two parts of different size: a long bit string B of length $n \cdot \mathrm{polylog}(m)$ and a small cached string C of length $\mathrm{poly}(\log m)$. Both these strings are prepared for a given set A of n elements in the universe of size m. When we need to answer a query "$x \in A$?", we use C to compute probabilistically a position in B and read one bit at this position. This is enough to answer whether x is an element of A, with a small one-sided error:

Theorem 19 ([j5]). *Fix any constant $\varepsilon > 0$. There exists a one-sided ε-error randomized scheme that includes a string B of length $O(n \log^2 m)$ and an auxiliary word C of length* $\mathrm{poly}(\log m)$. *We can answer membership queries "$x \in A$?" with one bit probe to B. For $x \in A$ the answer is always correct; for each $x \notin A$ the probability of error is less than ε. The position of the bit probed in A is computed from x and the auxiliary word C in time* $\mathrm{poly}(\log m)$.

Schemes with "cached" auxiliary information that depends on A (and not only on its size $n = \#A$ and the size m of the universe) make sense only if the cached information is very small. Indeed, if the size of the cached data is about $\log \binom{m}{n}$ bits,

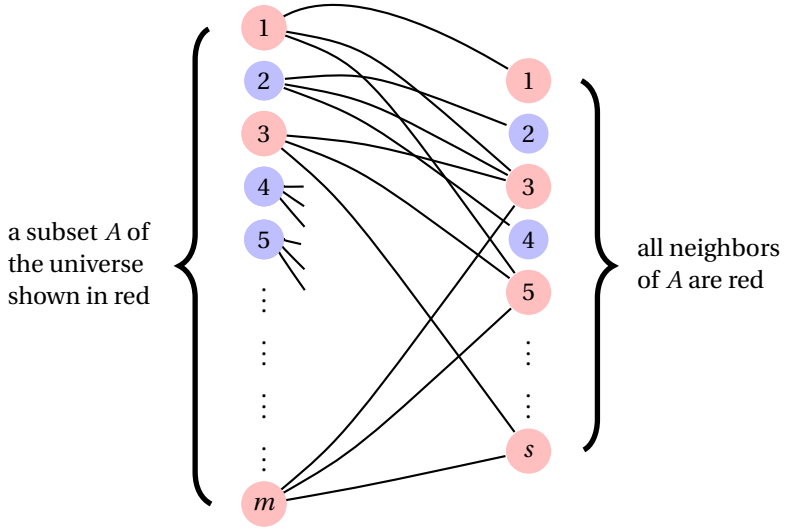

Figure 1.14: A bipartite graph representing a bit probe scheme.

then we can put there the list of all elements of A, so the problem becomes trivial. In Theorem 19 we use cached information of size $\text{poly}(\log m)$ bits. So this result is interesting when $\text{poly}(\log m) \ll n \ll m$, e.g., for $n = \exp\{\text{poly}(\log\log n)\}$. Note that due to Theorem 16 the space size $O(n\log^2 m)$ with one-sided error cannot be achieved by any schemes without cached auxiliary information that depends on A.

The model of data structures with cached memory looks useful for practical applications. The idea of splitting the data structure into "cached" and "remote" parts is very natural and quite common in computer science. Indeed, most practical computer systems contain a hierarchy of memory levels: CPU registers, several levels of processor caches, RAM, flash memory, magnetic disks, remote network-accessible drives, and so on. Each next level of memory is cheaper, but slower. So it is reasonable to study the trade-off between expensive and fast local memory, and cheap and slow external memory. However, this trade-off is typically studied for dynamic data structures. The same time, it is not obvious that fast cache memory of negligible size can help to process queries to a static data structure. Since a small cache "knows" virtually nothing about most objects in the database, at first sight, it seems to be useless. However, Theorem 19 shows that even a very small cache can be useful in static settings.

Let us sketch the construction used in Theorem 19. Our main data structure can be represented as a bipartite graph with m vertices in the left part (the size of the universe) and $s = O(n\log^2 m)$ vertices in the right part (the size of the database), see Fig. 1.14. The vertices of the graph are colored in red and blue. In the left part, we use the red color for elements in A and blue for others. In the right part, we use the red color for all neighbors of A and the blue color the other vertices.

When such graph is constructed, the queries are processed as follows. Given an element x from the universe, we take the xth vertex on in the left part of the graph and choose its random neighbor on the right. If the resulting vertex is red, we say that x belongs to A; otherwise (if the resulting vertex is blue) we say no. By construction, all neighbors of every vertex in A are red, so the answers for all $x \in A$ are always true. The same time, the vertices in the left part of the graph outside A can have both red and blue neighbors. The property of correctness of the scheme (for each $x \notin A$ the probability of a failure is less than ε) can be rephrased as follows:

> *For every vertex of the left part of the graph outside A, the fraction of red neighbors is less than ε.*

The nontrivial part of the proof is a construction of a graph with this property. It is rather easy to prove for every set A that such a graph exists. But we need a stronger statement: there exists a graph with the desired property with a concise description. The length of this description must be at most $\mathrm{poly}(\log m)$ bits. When such a graph is found, the coloring of its right part becomes the string B stored in the database, and a short description of the graph becomes the cached information C.

In the technical part of the argument (in a construction of a graph) we derandomize a probabilistic proof of the existence of graphs with the with required properties. In various publications, derandomization of probabilistic arguments involves highly sophisticated *ad-hoc* techniques. In our case, we do derandomization in a rather naive and straightforward way, following the ideas from [93] (on space-bounded Kolmogorov complexity). We take a suitable pseudo-random bits generator and construct the graph as an outcome of this generator. Thus, the required (concise) description of a graph is a suitable value of the seed of the generator.

To implement this plan, we need a generator such that with high probability (i.e., for most values of the seed) the resulting pseudo-random graph enjoys the required property. It turns out that several known generators fit our construction. Since the problem of testing the required property of a graph belongs to the computational class AC^0, we can use the Nisan–Wigderson generator or (thanks to the result of Braverman [83]) any polylog-independent function. Also the required property of a pseudo-random graph can be tested by a machine with logarithmic space. Hence, we can use Nisan's generator from [29]. We emphasize that our argument uses no unproven assumptions (like $\mathrm{P} \neq \mathrm{NP}$ or the existence of a one-way function).

In Theorem 19 we construct a scheme with an effective decoding: when the scheme is prepared, we can answer queries "$x \in A$?" in time polynomial in $\log m$. However the encoding (preparing the database and the auxiliary word for a given set A) runs in expected time $\mathrm{poly}(m)$. We assume that $n \ll m$, so the time polynomial in m seems to be too long. The next theorem claims that the encoding time can be reduced if we slightly increase the space of the scheme:

Theorem 20 ([j5])**.** *The scheme from Theorem 19 can be made effectively encodable in the following sense. Fix any constants $\varepsilon, \delta > 0$. There exists randomized scheme that includes a bit string B of length $n^{1+\delta}\mathrm{poly}(\log m)$ and an auxiliary word C of length* $\mathrm{poly}(\log m)$*. We can answer membership queries "$x \in A$?" with two bits probe to B. For $x \in A$ the answer is always correct; for $x \notin A$ probability of error is less than ε.*

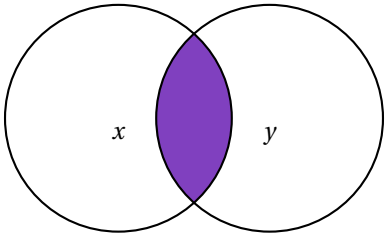

Figure 1.15: The mutual information for a pair of strings x, y.

The position of the bit probed in A is computed by x and the auxiliary word C in time $\mathrm{polylog}(m)$. *Given A, the entire scheme* (*the string B and the word C*) *can be computed probabilistically in average time* $\mathrm{poly}(n, \log m)$.

The proof of theorem 20 involves explicit constructions of expanders. For all technical detail, we refer the reader to [c6] and [j5] (see Appendix).

1.8 The Common Information of a Pair: Positive and Negative Results

In Section 1.6, we discussed a materialization of the quantity $\mathrm{C}(x \mid y)$ shown in the diagram in Fig. 1.13. In this section we deal with a materialization of the quantity shown in the diagram in Fig. 1.15, which corresponds to the mutual information of two strings. In what follows we briefly discuss the notion of *common information* and several results on the common information of a pair of strings.

Intuitively, the common information of strings x and y is thought as an "extraction" of the information shared by these two strings. In some sense, the common information is a more material version of the mutual information $I(x : y)$. Speaking very informally we say that a string w is a common information extracted from strings x and y, if w can be easily computed from x and likewise from y. It remains to make more precise the idea of "easy computing." In this context it is usually assumed that we may obtain w from x and from y using a few help bits. Typically it is required that $\mathrm{C}(w \mid x) = O(\log n)$ and $\mathrm{C}(w \mid y) = O(\log n)$, where n is the length of x and y, and the constant hidden in the $O(\cdot)$ notation depends only on the universal machine. In a more liberal setting we say that $\mathrm{C}(w \mid x) = \alpha$ and $\mathrm{C}(w \mid y) = \beta$ for some integer α and β, and then study the trade-off between the parameters α, β and the maximum of $\mathrm{C}(w)$. From (1.11) it follows that for small α, β the value of $\mathrm{C}(w)$ cannot be much larger than $I(x : y)$ (i.e., the *common information* is not larger than the *mutual information*). The question is whether the common information can approach the value of the mutual information. In other words, we ask when the whole mutual information can be "materialized."

The notion of common information appeared in 1973 in [11], where P. Gács and J. Körner discussed this information quantity in two frameworks — for Shannon's

entropy and for Kolmogorov complexity. Gács and Körner showed that for some strings x and y the common information is much smaller than the mutual information. The notion of common information for Kolmogorov complexity was revisited in the 1990s, see [43, j17, j15]. These works offered a subtler analysis of the possible size of the common information for different regimes of parameters and suggested several explicit examples of pairs x, y for which the common information is much smaller than mutual information. A self-contained exposition of these results can be found in [99, Chapter 11].

In what follows, we proceed with several results on extracting the common information (with those that are used in subsequent sections). We start with a formal definition.

Definition 6. *We say that k bits of common information can be extracted from x and y with inaccuracies (α, β), if there exists a string w such that*

$$\begin{cases} C(w \mid x) & \le & \alpha, \\ C(w \mid y) & \le & \beta, \\ C(w) & \ge & k. \end{cases}$$

Denote by $\mathrm{ComInf}(x, y)$ *the set of all triples (k, α, β) such that k bits of common information can be extracted from x and y, with inaccuracies (α, β).*

It is known that $\mathrm{ComInf}(x, y)$ is not uniquely defined by the complexity profile of (x, y). That is, pairs (x, y) and (x', y') with very similar (or even identical) complexity profiles can have very different properties of "extractability" of the mutual information, see [j15]. The next theorem describes the minimal possible set of triples $\mathrm{ComInf}(x, y)$ for a pair with one specific complexity profile.

Theorem 21. *[43] For every $\delta > 0$ and for all large enough n there exists a pair of strings (x, y) such that $C(x) =^+ 2n$, $C(y) =^+ 2n$, $C(x, y) =^+ 3n$, and* no *triple (k, α, β) satisfying*

$$\begin{cases} \alpha \le (1-\delta)n, \\ \beta \le (1-\delta)n, \\ \alpha + \beta + \delta n \le k \end{cases}$$

belongs to $\mathrm{ComInf}(x, y)$.

In particular, for x, y from Theorem 21, we know that if $\mathrm{C}(w \mid x)$ and $\mathrm{C}(w \mid y)$ are very small, than the value of $\mathrm{C}(w)$ must be also very small, though the mutual information between x and y is pretty large, $I(x : y) =^+ n$.

Remark 6. Theorem 21 can be made stronger: in all inequalities, the terms of size (δn) can be replaced by $\Theta(\log n)$, with coherently chosen constant factors. We use a slightly weaker version of the theorem to simplify the notation.

Remark 7. The condition suggested in Theorem 21 is in a sense optimal. If we make the constraints slightly weaker, then we obtain a property of extractability of the common information that holds true for all pairs of strings, see [j15] for details.

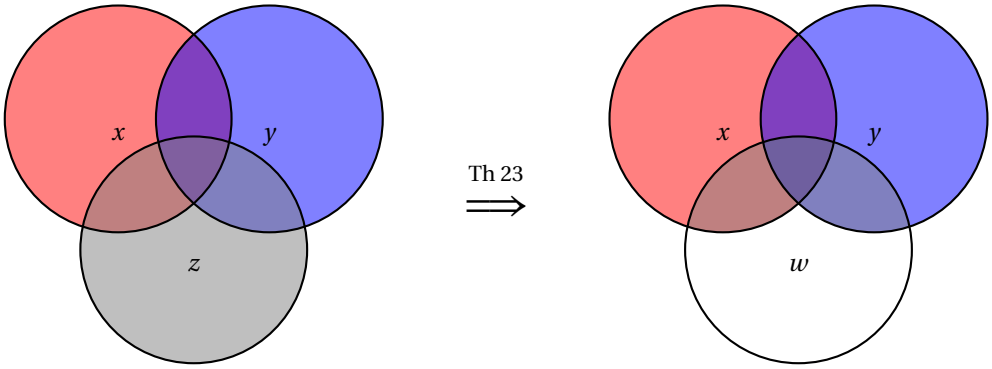

Figure 1.16: A string z helps extract the common information from x, y. Areas of the same color (on the left and on the right) represent equal complexity quantities. The vanishing information quantity is shown in white.

I. Razenshteyn proposed a more constructive version of Theorem 21. We cannot construct algorithmically a pair (x, y) satisfying Theorem 21 since there is obviously no short algorithmic description of any object of high Kolmogorov complexity. However, we can provide a large constructive set where the majority of elements have the required property:

Theorem 22 (I. Razenshteyn, [86])**.** *For every $\delta > 0$ there exists an algorithm that for all large enough n generates a list of pairs S_n of size $2^{3n+O(\log n)}$ such that for the majority of $(x, y) \in S_n$ it holds $C(x) =^+ 2n$, $C(y) =^+ 2n$, $C(x, y) =^+ 3n$, and* no *triple (k, α, β) satisfying*

$$\begin{cases} \alpha \le (1-\delta)n, \\ \beta \le (1-\delta)n, \\ \alpha + \beta + \delta n \le k \end{cases}$$

belongs to ComInf(x, y).

Remark 8. Technically [86] contains Theorem 22 only in the symmetric setting $\alpha = \beta$. However, Razenshteyn's proof applies in the general case.

When we claim in Theorem 22 that an algorithm generates a set S_n, we mean that on the input n the algorithm prints the list of elements of this set and stops, i.e., the enumeration of S_n has an explicit and distinguishable completion. In other words, the pairs (x, y) from Theorem 22 are $(O(\log n), O(\log n))$-stochastic in the sense of Definition 4. Thus, Theorem 22 claims that there are $(O(\log n), O(\log n))$-stochastic pairs with the worst possible property of extractability of the common information.

We proceed with a positive result on extracting the common information. It applies only to stochastic pairs (x, y). The next theorem allows to extract the common information from a stochastic pair (x, y) assuming the existence of another string z, with some specific conditions for the joint complexity profile of x, y, z.

Theorem 23 (A special case of a more general theorem proven in [j9])**.** *For every* $(O(\log n), O(\log n))$*-stochastic pair of strings* (x, y) *and for every* z *there exists another string* w *such that*

$$\begin{array}{lll} I(x:w \mid y) & =^+ & I(x:z \mid y), \\ I(y:w \mid x) & =^+ & I(y:z \mid x), \\ I(x:y:w) & =^+ & I(x:y:z), \\ \mathrm{C}(w \mid x,y) & =^+ & 0, \end{array}$$

where n *is* $\mathrm{C}(x,y,z)$.

Remark 9. From the four equation in Theorem 23 it follows that $\mathrm{C}(w) =^+ I(x,y:z)$.

Remark 10. The constant factors in the $O(\log n)$-terms hidden in the notation $=^+$ used in this theorem depend implicitly on the constant from the $O(\cdot)$ terms in the definition of $(O(\log n), O(\log n))$-stochasticity.

The statement of Theorem 23 is illustrated in Fig. 1.16: when substituting z by w we keep untouched all atomic information quantities for the triple (x,y,z) except for the vanishing $\mathrm{C}(z \mid x,y)$.

Corollary 2. *If a pair of strings* (x,y) *is* $(O(\log n), O(\log n))$*-stochastic, and there exists a string* z *such that* $I(x:z \mid y) =^+ 0$ *and* $I(y:z \mid x) =^+ 0$*, then at least* $I(x:y:z)$ *bits of common information can be extracted from* x *and* y *with inaccuracies* $O(\log n)$.

If in addition $I(x:y \mid z) =^+ 0$*, then the whole mutual information* $I(x:y)$ *can be extracted from* x *and* y *with inaccuracies* $O(\log n)$.

This corollary can be obtained by specializing Theorem 23 to the case when two or all three quantities of the conditional mutual information for x,y,z are negligibly small, see Fig. 1.17

Proof of Theorem 23: Step 1 [Constructing the set of clones of (x,y).] By definition of stochasticity, (x,y) belongs to a set S such that $\mathrm{C}(S) = O(\log n)$ and

$$\mathrm{C}(x,y) =^+ \log \# S.$$

Denote A_0 the set of all "clones" of (x,y), i.e., the set of all pairs that have similar complexity profiles with respect to z. More precisely, le A_0 be the set of pairs $(x',y') \in S$ such that all complexity quantities involving x', y', and having z as a condition are not greater than the corresponding complexity quantities for x and y, i.e.,

$$\begin{array}{l} \mathrm{C}(x' \mid z) \le \mathrm{C}(x \mid z), \\ \mathrm{C}(y' \mid z) \le \mathrm{C}(y \mid z), \\ \mathrm{C}(x',y' \mid z) \le \mathrm{C}(x,y \mid z), \\ \mathrm{C}(x' \mid y',z) \le \mathrm{C}(x \mid y,z), \\ \mathrm{C}(y' \mid x',z) \le \mathrm{C}(y \mid x,z). \end{array}$$

It is not hard to show (see [j9] for detail) that the sizes of sections and projections of A_0 are bounded by the exponents of the corresponding complexity quantity for x,y conditional on z. For example, the cardinality of the whole A_0 is not greater than $2^{\mathrm{C}(x,y|z)+1}$, the cardinality of the projection of A_0 onto the first coordinate is not

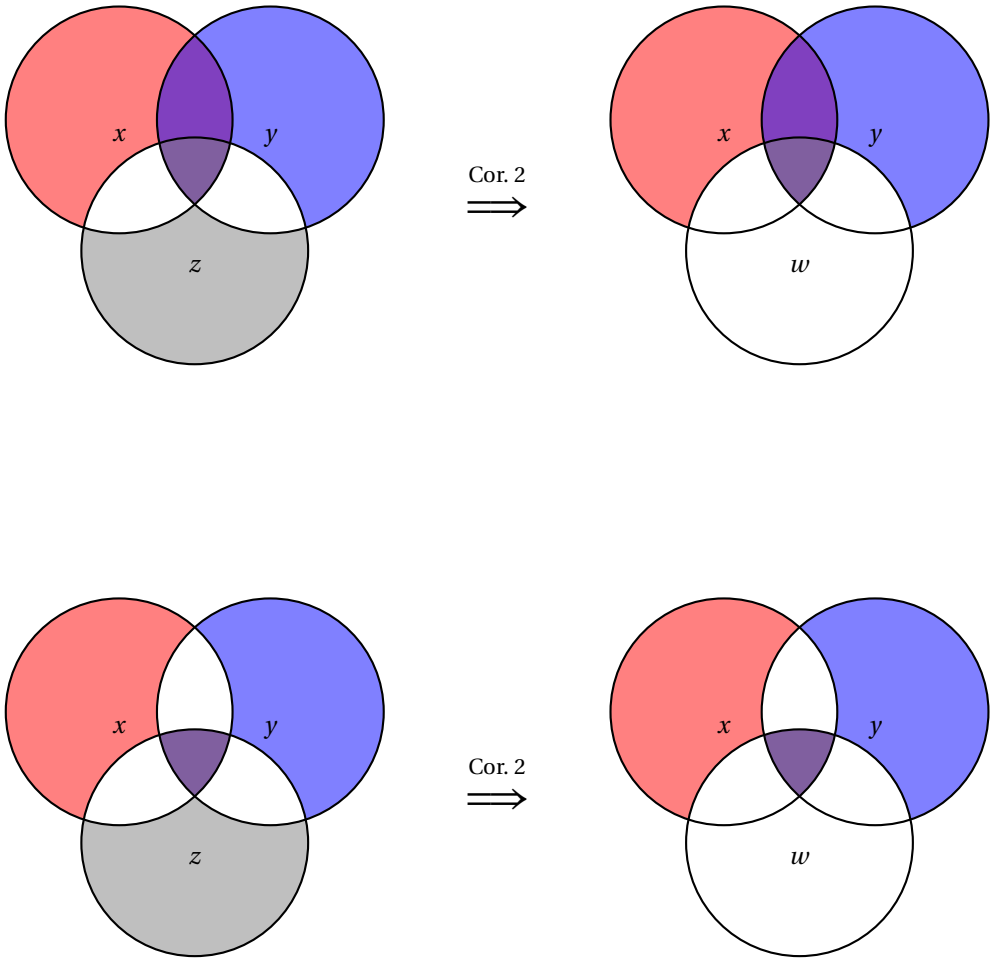

Figure 1.17: A string z helps extract the common information from x, y, see Corollary 2. Areas of the same color (on the left and on the right) represent equal complexity quantities. The vanishing information quantities are shown in white.

greater than $2^{C(x|z)+1}$, for every x' the number of strings y' such that $(x', y') \in A_0$ is not greater than $2^{C(y|x,z)+1}$, and so on.

We will say that a positive integer N *is not greater than* M *in order of magnitude* if the binary expansion of N is not longer (measured in bits) than the binary expansion of M. Obviously, if N is not greater than M in order of magnitude, then $N < 2M$. Note that that the set of all N that are not greater than M in order of magnitude can be specified $\log M$ bits (we only need to know the number of digits in the binary expansion of M).

We say that a set $A \subset S$ is *valid* if the cardinalities of all its sections and projections are not greater in order of magnitude than the the cardinalities of the corresponding sections and projections of A_0. (In particular, the set A_0 is valid.)

Given the values of all complexity quantities involving (x, y, z) and the list of all elements of S, we can algorithmically find all valid sets (the list of the valid sets is huge, but it is finite). Since the Kolmogorov complexity of the list of all elements of S is logarithmic in n, we conclude that the list of all valid sets also has complexity $O(\log n)$. Denote by $A_1, A_2, \ldots$ the lexicographically ordered list of all valid sets. Note that by definition the size of a valid set is at most twice as large as the size of A_0. Similar bounds hold for the cardinalities of sections and projections of each valid set.

Step 2 [Selecting well-behaved clones]. We select from the list of valid sets some "special" subsequence. We do it inductively as follows. Assume that the first $(s-1)$ valid sets in the list $A_1, \ldots, A_{s-1}$ has been examined, and $A_{i_1}, \ldots, A_{i_k}$ has been selected as special. Then we examine the next valid set A_s; we select A_s as special if the difference

$$A_s \setminus \left(\bigcup_{r \le k} A_{i_r} \right)$$

contains at least $2^{C(x,y|z)-\kappa \log n}$ pairs (κ to be specified below). In other words, we select A_s if it brings many new pairs that have not been covered by A_{i_r} selected earlier.

By construction the special subsequence contains at most

$$\frac{\#S}{2^{C(x,y|z)-\kappa \log n}} = 2^{I(x,y:z)+\kappa \log n + O(\log n)} \tag{1.19}$$

valid sets. Denote by $\hat{A}$ the union of all selected valid sets.

Note that $C(\hat{A}) = O(\log n + \log \kappa)$, since the lists of elements of these sets can be found algorithmically given the complexity values involving x, y, z, the list of elements of S, and the value of κ.

Lemma 2. *The pairs* (x, y) *belongs to* $\hat{A}$ (*assuming that* κ *is large enough*).

Proof of lemma: Assume for the sake of contradiction that (x, y) does not belong to $\hat{A}$. This means that A_0 (which is a valid set) was not included in the selected subsequence of valid sets. It follows that the cardinality of the difference $A_0 \setminus \hat{A}$ is less than $2^{C(x,y|z)-\kappa \log n}$.

Given z, we can specify the pair (x, y) as an element of $A_0 \setminus \hat{A}$. To this end, we need to know the complexity quantities involving x, y, z, the value of κ, and the ordinal

number of the pair (x, y) in the natural enumeration of $A_0 \setminus \hat{A}$. We obtain

$$\begin{aligned} \mathrm{C}(x, y \mid z) &\le \log \#(A_0 \setminus \hat{A}) + O(\log n + \log \kappa) \\ &\le \mathrm{C}(x, y \mid z) - \kappa \log n + O(\log n + \log \kappa). \end{aligned}$$

We get a contradiction be choosing a large enough κ. □

Remark 11. In Lemma 2 we can choose a constant value of κ that does not depend on n.

Step 3 [Retrieving w from the typical clones.] We conclude from Lemma 2 that (x, y) belongs to some selected sets A_i. Denote by w the index of the first selected valid set covering (x, y). By the definition of A_w (due to the bounds on the cardinalities of sections and projections in every valid set) we have

$$\begin{aligned} &\mathrm{C}(x, y \mid w) \le^+ \mathrm{C}(x, y \mid z), \\ &\mathrm{C}(x \mid w) \le^+ \mathrm{C}(x \mid z), \\ &\mathrm{C}(y \mid w) \le^+ \mathrm{C}(y \mid z), \\ &\mathrm{C}(x \mid y, w) \le^+ \mathrm{C}(x \mid y, z), \\ &\mathrm{C}(y \mid x, w) \le^+ \mathrm{C}(y \mid x, z). \end{aligned}$$

Further, from the bound on the number of selected sets (1.19) we obtain

$$\mathrm{C}(w) \le^+ I(x, y : z).$$

These inequalities (together with the values of $\mathrm{C}(x)$, $\mathrm{C}(y)$, and $\mathrm{C}(x, y)$) uniquely determine all information quantities involving (x, y, w). It is not hard to verify that the value of $\mathrm{C}(w \mid x, y)$ vanishes and all other "atomic" information quantities for (x, y, w)

$$\mathrm{C}(x \mid y, w),\ \mathrm{C}(y \mid x, w),\ I(x : y \mid w),\ I(x : w \mid y),\ I(y : w \mid x),\ I(x : y : w)$$

are equal (up to a logarithmic term) to the corresponding values for (x, y, z),

$$\mathrm{C}(x \mid y, z),\ \mathrm{C}(y \mid x, z),\ I(x : y \mid z),\ I(x : z \mid y),\ I(y : z \mid x),\ I(x : y : z)$$

respectively, as shown in Fig. 1.17. □

Remark 12. We stated and proved Theorem 23 for $(O(\log n), O(\log n))$-stochastic pairs (x, y). This result can be easily extended to stochastic k-tules of strings for any $k > 2$. More precisely, we can prove (with essentially the same argument as above) that for every $(O(\log n), O(\log n))$-stochastic tuple of strings $(x_1, \ldots, x_k)$ and for every string z, there exists a string w such that all "atomic" complexity quantities for $(x_1, \ldots, x_k, w)$ are the same as the corresponding quantities for $(x_1, \ldots, x_k, z)$ with one exception: whatever $\mathrm{C}(z \mid x_1, \ldots, x_k)$ is, we guarantee that $\mathrm{C}(w \mid x_1, \ldots, x_k) =^+ 0$ (the value of n hidden in the logarithmic terms is $\mathrm{C}(x_1, \ldots, x_k, z)$).

In [j9] we proved a somewhat more general statement, which was used in a study of the behavior of Kolmogorov complexities in the relativization with a random oracle.

Theorem 23 can be viewed as a version of the Ahlswede–Körner lemma from Shannon's information theory (see Theorem 2 in [14] and [64]; a similar technique was used, e.g., in [17]; see also a discussion of *common information* in [85, Section "*Source and Channel Networks*"]) adapted to Kolmogorov complexity. It remains unknown whether the condition of stochasticity of (x, y) can be omitted in Theorem 23 and in Corollary 2.

1.9 From extracting common information to non-classic information inequalities

In this section we show how the technique of extracting the common information helps prove new (non-Shannon-type) information inequalities. The parallels between Shannon's entropy and Kolmogorov complexity appears to be a crucial clue in this direction.

We start with a simple and pretty standard fact from Shannon's version of information theory.

Lemma 3 (see the textbook by I. Csiszar and J. Körner [85])**.** *Let X, Y, Z be random variables such that*

$$I(X : Y \mid Z) = I(X : Z \mid Y) = I(Y : Z \mid X) = 0. \tag{1.20}$$

Then there exists a random variable W that "materializes" the mutual information of X, Y, and Z in the sense that

$$H(W \mid X) = H(W \mid Y) = H(W \mid Z) = 0,$$

and X, Y, Z are independent conditional on W.

Sketch of proof: Fist of all we observe that for X, Y, Z satisfying (1.20)

$$I(X : Y) = I(X, Y : Z) = I(X : Y : Z),$$

see Fig. 1.18. Let us split the values of the pair (X, Y) in equivalence classes so that equivalent values correspond to identical conditional distributions on Z. Define W as the index of the equivalence class corresponding to the given (randomly chosen) values of (X, Y). Observe that $I(W : Z) = I(X, Y : Z)$ since W contains the same information about Z as the pair (X, Y). Hence,

$$H(Z) \geq I(X : Y : Z).$$

The conditions $I(X : Z \mid Y) = 0$ and $I(Y : Z \mid X) = 0$ imply that W can be obtained deterministically from X and from Y (the value of only X or only Y is enough to know the distribution on Z conditional on (X, Y)), so $H(W \mid X) = H(W \mid Y) = 0$. Therefore, we can apply a version of (1.11) for Shannon's entropy

$$H(W) \leq H(W \mid X) + H(W \mid Y) + I(X : Y)$$

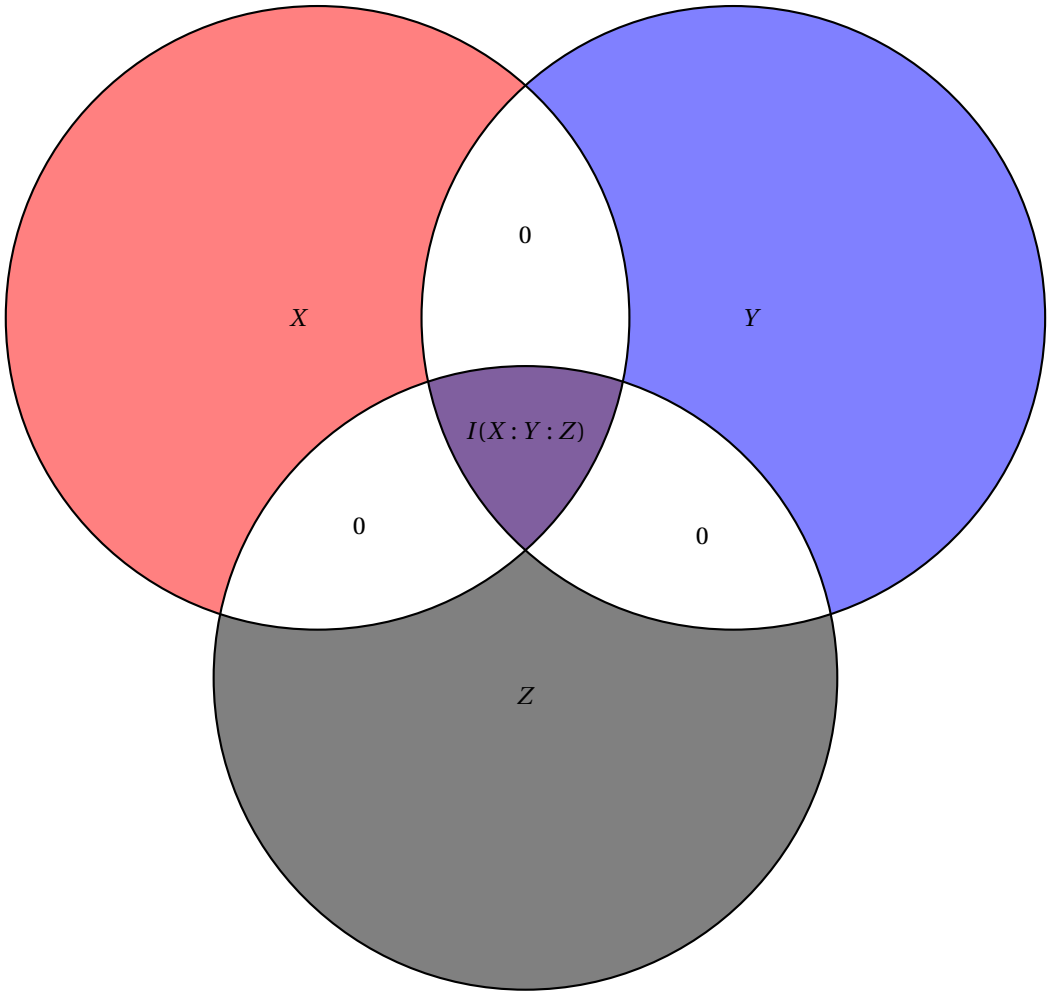

Figure 1.18: A triple of random variables with vanishing conditional mutual information.

and obtain $H(W) \le I(X : Y : Z)$. Then we use another version of (1.11) for the conditional distributions,

$$H(W \mid Z) \le H(W \mid X) + H(W \mid Y) + I(X : Y \mid Z),$$

and conclude that $H(W \mid Z) = 0$. It is easy to verify that X, Y, and Z are independent conditional on W. □

We use this lemma to prove a "constraint" inequality for Shannon's entropy:

Proposition 1. *For any tuple of jointly distributed random variables A, B, X, Y, Z, if (X, Y, Z) satisfies* (1.20) *then*

$$I(X : Y) \le I(X : Y \mid A) + I(X : Y \mid B) + I(A : B). \tag{1.21}$$

Sketch of proof: We know from Lemma 3 that there exists a W that materializes the mutual information between X, Y, Z. By applying the version of (1.11) we obtain

$$H(W) \le H(W \mid A) + H(W \mid B) + I(A : B).$$

From the definition of W it easily follows that $H(W) = I(X : Y)$, and the conditional entropies $H(W \mid A)$ and $H(W \mid B)$ coincide with $I(X : Y \mid A)$ and $I(X : Y \mid B)$ respectively. □

It is known that inequality (1.21) is not valid for Shannon's entropy in general (for the distributions that do not satisfy (1.20)), see [j16]. A natural question arises: is there a more "robust" version of this constraint inequality that remains valid if (1.20) is satisfied up to a "small enough error"? More specifically, if

$$I(X : Y \mid Z),\ I(X : Z \mid Y),\ I(Y : Z \mid X) \le \varepsilon,$$

can we conclude that

$$I(X : Y) \le I(X : Y \mid A) + I(X : Y \mid B) + I(A : B) + O(\varepsilon)$$

for all A and B?

Apparently Lemma 3 does not apply when (1.20) is not true precisely, so an "error-tolerant" version of Proposition 1 requires a new proof technique. We will see that Kolmogorov complexity is quite appropriate in this context.

We try to transpose the proof of Proposition 1 to the framework of Kolmogorov complexity. First of all, we translate the statement of Lemma 3 in the language of Kolmogorov complexity. For Kolmogorov complexity there is no way to say that the value of the mutual information is exactly zero, we only can say that it is negligibly small. With logarithmic inaccuracy (which seems to be the most natural technical version of "negligibly small" quantity for Kolmogorov complexity) we can conjecture the following proposition.

Plausible Statement. *Let x, y, z be binary strings such that*

$$I(x : y \mid z) =^+ I(x : z \mid y) =^+ I(y : z \mid x) =^+ 0.$$

Then there exists a string w that "materialize" the mutual information of x, y, and z in the sense that

$$\mathrm{C}(w \mid z) =^+ \mathrm{C}(w \mid y) =^+ \mathrm{C}(w \mid z) =^+ 0,$$

and x, y, z are independent conditional on w, i.e.,

$$\mathrm{C}(x, y, z \mid w) =^+ \mathrm{C}(x \mid w) + \mathrm{C}(y \mid w) + \mathrm{C}(z \mid w).$$

This plausible statement is true, and we prove it in the next section. Now we are not ready to prove it for all x, y, z. However, we can prove it immediately for $(O(\log n), O(\log n))$-stochastic triples. Indeed, for all stochastic tuples this statement follows from Theorem 23, see Corollary 2. Now we can reproduce the proof of Proposition 2 in the framework of Kolmogorov complexity, which results in the following proposition:

Proposition 2. *For any tuple of $(O(\log n), O(\log n))$-stochastic strings (a, b, x, y, z), if (x, y, z) satisfy* (1.22) *then*

$$I(x : y) \leq^+ I(x : y \mid a) + I(x : y \mid b) + I(a : b). \tag{1.22}$$

The parameter n in all logarithmic terms is $\mathrm{C}(a, b, x, y, z)$.

Proposition 2 is already an interesting result, but let us go further. Can we adapt the argument to a weaker version of (1.22), where the values of the conditional mutual information are bounded by εn? Can we establish a trade-off between the inaccuracy in (1.22) and the resulting inaccuracy in (1.22)?

To address these questions, we take the general version of Theorem 23 (instead of the special case from Corollary 2) and substitute it into the proof of (1.21) from Proposition 1.21. The result can be stated as follows:

Theorem 24. *For any $(O(\log n), O(\log n))$-stochastic triple (x, y, z) and for any a, b, c*

$$\begin{aligned} I(x : y) \leq^+ & I(x : y \mid a) + I(x : y \mid b) + I(a : b) \\ & + I(x : y \mid z) + I(x : z \mid y) + I(y : z \mid x), \end{aligned} \tag{1.23}$$

where n in all logarithmic terms is $\mathrm{C}(a, b, x, y, z)$.

(When given Theorem 23, the inference of (1.23) is quite straightforward, see [j13] for technical details.) Theorem 5 and Theorem 7 allow to prove (1.23) for Kolmogorov complexity of all tuples (not necessary stochastic) and for Shannon's entropy:

Theorem 25 ([j13])**.** *(a) Inequality* (1.23) *is true for all tuples of strings (a, b, x, y, z).*
(b) For all jointly distributed random variables (A, B, X, Y, Z)

$$\begin{aligned} I(X : Y) \leq & I(X : Y \mid A) + I(X : Y \mid B) + I(A : B) \\ & + I(X : Y \mid Z) + I(X : Z \mid Y) + I(Y : Z \mid X). \end{aligned} \tag{1.24}$$

This theorem directly implies an error-tolerant version of Proposition 1:

Corollary 3. *For all jointly distributed random variables* (A, B, X, Y, Z) *and for every* $\varepsilon > 0$*, if the quantities* $I(X : Y \mid Z)$*,* $I(X : Z \mid Y)$*,* $I(Y : Z \mid X)$ *are not greater than* ε*, then*

$$I(X : Y) \le I(X : Y \mid A) + I(X : Y \mid B) + I(A : B) + 3\varepsilon.$$

This corollary closes the circle: we have obtained a positive answer to the initial question stated at the beginning of this section.

Inequality (1.24) is *non-Shannon-type*, i.e., it cannot be obtained as a convex combination of the standard inequalities of monotonicity, subadditivity, and submodularity (see [j13] for the proof). Specialization of (1.24) to $A = Z$ gives exactly the very first example of a non-Shannon-type inequality from [44].

Applying a more general version of Theorem 23 (see Remark 12) we can deduce a countable sequence of non-Shannon-type information inequalities for Kolmogorov complexities of k-tuples of binary strings (or entropies of k-tuples of random variables) with $k > 4$, see [j13].

Historical remark. The original proof of a non-Shannon-type inequality in [44] used the method of reducing the conditional mutual information, which is pretty different from the argument used in this section. The core of this method can be expressed in terms of *adhesive extensions*, see [72]. A somewhat more general version of this framework (so-called *book extensions*) is presented in [92]. The proof of (1.24) with the technique of extracting the mutual information was suggested in [j13]. This argument was presented in [j13] in the language of Shannon's entropy, with the original version of the Ahlswede-Körner lemma instead of Theorem 23 (though an application to Kolmogorov complexity was the primary motivation).

Variations of these techniques remain until now the only known tools for deducing non-Shannon-type inequalities. T. Kaced showed in [90] that the standard versions of these techniques are equivalent (a proof with one method can be transformed in a proof with another one and visa-versa). Both techniques can be represented in the framework of self-adhesive polymatroids, see [97].

1.10 From Non-Classic Information Inequalities to Extracting Common Information

In the previous section, we used the materialization of the mutual information to prove non-Shannon type information inequalities. In this section, we go in the opposite direction and use a non-Shannon type information inequality to materialize the mutual information. Technically, we prove the Plausible Statement on p. 65 from the previous section. Let us formulate the statement more precisely, with explicit quantifiers for all constant factors.

Theorem 26. *[j12] For every* $C_1 > 0$ *there exists a* $C_2 > 0$ *such that for all of binary strings* x, y, z*, if the values* $I(x : y \mid z)$*,* $I(x : z \mid y)$*, and* $I(y : z \mid x)$ *are less than* $C_1 \log n$*, then the mutual information between* x*,* y*, and* z *can be extracted with inaccuracy*

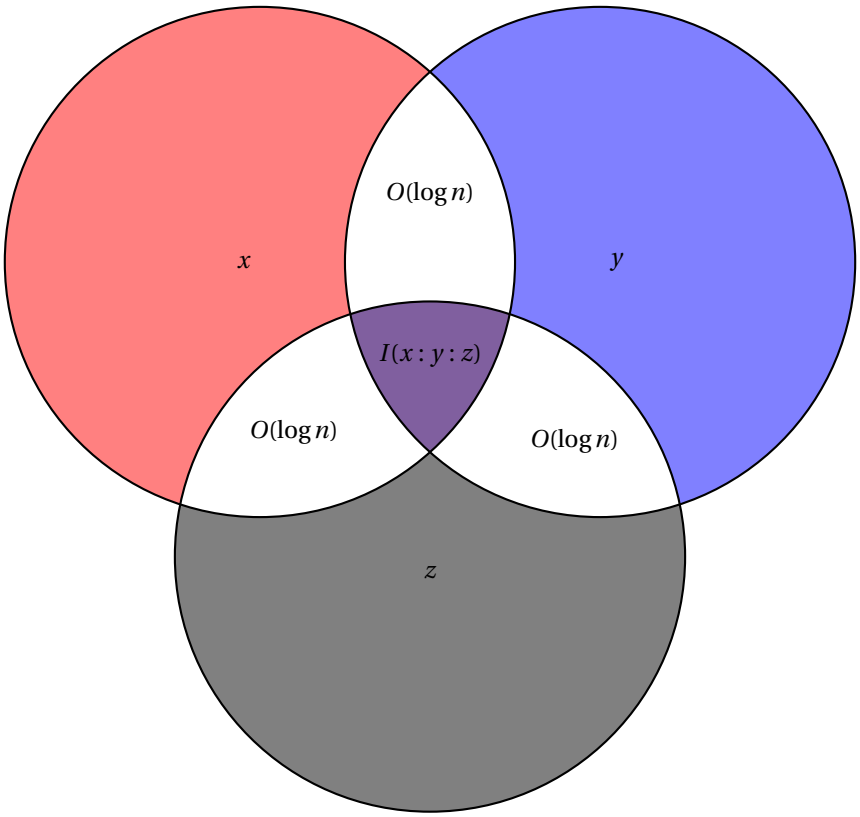

Figure 1.19: A triple of strings with negligibly small conditional mutual informations.

$C_2 \log n$, i.e., there exists a string w such that

$$\begin{array}{rcl} \mathrm{C}(w \mid x) & \leq & C_2 \log n, \\ \mathrm{C}(w \mid y) & \leq & C_2 \log n, \\ \mathrm{C}(w \mid z) & \leq & C_2 \log n, \\ \mathrm{C}(w) & \geq & I(x:y:z) - C_2 \log n. \end{array}$$

Here, n denotes the Kolmogorov complexity of the given triple of strings, $n = \mathrm{C}(x,y,z)$.

Under the condition

$$I(x:y \mid z) =^+ I(x:z \mid y) =^+ I(y:z \mid x) =^+ 0$$

we have

$$I(x:y:z) =^+ I(x:y) =^+ I(x:z) =^+ I(y:z),$$

see the diagram in Fig. 1.19. Assuming that

$$\mathrm{C}(w \mid x) =^+ \mathrm{C}(w \mid y) =^+ \mathrm{C}(w \mid z) =^+ 0,$$

we can conclude that $C(w)$ is not greater than the quantities of the mutual information $I(x:y:z)$. Hence, the bound $C(w) \geq^+ I(x:y:z)$ in Theorem 26 is tight (up to $O(\log n)$).

It is interesting to compare the problem of materializing of the mutual information for pairs and triples of string. We say, somewhat informally, that *the mutual information between x and y can be completely materialized* if there exists a w such that $C(w \mid x) =^+ 0$, $C(w \mid y) =^+ 0$ (w is a common part of x and y), and

$$C(x, y \mid w) =^+ C(x \mid w) + C(y \mid w)$$

(x and y are independent conditional on w, i.e., the whole mutual information between x and y is included in w). Similarly, we say that *the mutual information between x, y, and z can be completely materialized* if there exists a w such that $C(w \mid x) =^+ 0$, $C(w \mid y) =^+ 0$, $C(w \mid z) =^+ 0$, and

$$C(x, y, z \mid w) =^+ C(x \mid w) + C(y \mid w) + C(z \mid w).$$

Observe that this condition holds true for w from Theorem 26. We have mentioned in Section 1.8 that the possibility to completely materialize the mutual information between x and y is not determined by the complexity profile of (x, y). Theorem 26 implies that for triples of strings the situation is drastically different: the mutual information between x, y, and z can be completely materialized if and only if the quantities $I(x:y \mid z)$, $I(x:z \mid y)$, and $I(y:z \mid x)$ are negligibly small.

In the rest of this section, we prove Theorem 26. Our argument is based on the technique of "bunches" suggested in [j12].

Definition 7. *An* (α, β, γ)-bunch X *is a set of strings such that*

1. $\#X = 2^\alpha$,
2. $C(x_1 \mid x_2) < \beta$ *for all* $x_1, x_2 \in X$,
3. $C(x) < \gamma$ *for all* $x \in X$.

Remark 13. The property of being an (α, β, γ)-bunch is enumerable.

Example. We construct an example of an $(n, n + O(1), 2n + O(1))$-bunch as follows. Let us fix some string w of length n and define X as the set of all strings x of length $2n$ having w as a prefix. In other words, all strings in X share the same first n bits (taken from w), and only the last n bits vary. Then for all strings x, x' in this set we have $C(x \mid x') \leq n + O(1)$, so the definition of a bunch applies. We recommend to the reader to keep in mind this simple example (a bunch with parameters $(\approx n, \approx n, \approx 2n)$) in all subsequent arguments in this section.

This example suggests that an (α, β, γ)-bunch can be though as a "sunflower" with some core w and 2^α "petals," where each petal adds at most β bits of new information to the core. The next proposition claims that in fact every bunch has essentially this structure of a sunflower.

Proposition 3. *For every (α,β,γ)-bunch X there exists a string w such that* $\mathrm{C}(w) \le^+ \gamma-\alpha+\epsilon$ *and for every $x \in X$*

$$\mathrm{C}(x \mid w) \le^+ \beta+\epsilon,$$

where $\epsilon = \beta-\alpha+O(1)$.

Intuitively, this proposition claims that we can find a "core" in every bunch. The statement is interesting when $\alpha \approx \beta$. The proof of this proposition uses the following lemma.

Lemma 4. *There exists an algorithm that takes a triple of integers α,β,γ as an input, and enumerates a list of (α,β,γ)-bunches $U_0,\dots,U_q$ such that:*

- *for every (α,β,γ)-bunch U there exists $i \le q$ such that $\#(U \cap U_i) \ge 2^{\alpha-\epsilon}$,*
- *$q < 2^{\gamma-\alpha+\epsilon}$,*

where $\epsilon = \beta-\alpha+O(1)$.

Proof of Proposition 3. We apply Lemma 4 and take a bunch U_i (selected for the enumeration in the lemma) such that $\#(U \cap U_i) \ge 2^{\alpha-\epsilon}$. We set w to be the index of U_i. By Lemma 4 we have $i \le 2^{\beta+\gamma-2\alpha+O(1)}$, so we conclude that

$$\mathrm{C}(w) \le^+ \beta+\gamma-2\alpha.$$

It remains to estimate $\mathrm{C}(x \mid w)$ for x in the bunch U. We know that for every $a \in U \cap U_i$

$$\mathrm{C}(a \mid w) \le^+ \alpha \tag{1.25}$$

(since a belongs to U_i), and

$$\mathrm{C}(x \mid a) < \beta \tag{1.26}$$

(since a belongs to U). In other words, there are at least $\#(U \cap U_i)$ two-step chains

$$w \to a \to x$$

that satisfy (1.25) and (1.26). The cardinality of the set of x with this property is not greater than

$$\frac{2^\alpha \cdot 2^\beta}{\#(U \cap U_i)} \le 2^{\beta+\epsilon},$$

and the set of such x is enumerable (given w and the integers α,β,γ). To find x given w as a preliminary condition, we need to know the index of x in this enumeration. It follows that

$$\mathrm{C}(x \mid w) \le^+ \beta+\epsilon,$$

and we are done. □

Proof of Lemma 4: Fix an algorithm that takes a triple of integers α, β, γ as an input and enumerates the list of all (α, β, γ)-bunches. Though the number of bunches with given parameters is finite, this enumeration has no distinguishable completion (we can never decide whether all bunches are already found). We call this algorithm *simple enumerator*.

Now we define another algorithm that *selects* some subsequence from the list of all bunches generated by the simple enumerator. The selection is done by the following rule. Assume that bunches $U_0, \dots, U_s$ are already *selected*; let the simple enumerator return another bunch V. Denote $\Delta = \beta - \alpha + 2$. If $\#(V \cap U_i) < 2^{\alpha-\Delta}$ for every $i = 0, \dots, s$, then we *select* this bunch and let $U_{s+1} = V$. Otherwise we skip V and wait for the next bunch from the simple enumerator.

Let $U_0, \dots, U_q$ be the list of all bunches that are selected in this procedure. From the construction it follows that for every bunch V either $V = U_i$ or at least $\#(V \cap U_i) \geq 2^{\alpha-\Delta}$ (for at least one selected bunch U_i). Also by construction, $\#(U_i \cap U_j) < 2^{\alpha-\Delta}$ for every two *selected* bunches U_i, U_j. It remains to show that the number of selected bunches is not too large.

It is enough to prove that every string x belongs to less than $2^{\beta-\alpha+1}$ bunches. Indeed, there are less than 2^γ strings x such that $\mathrm{C}(x) < \gamma$. If every x is covered by at most $2^{\beta-\alpha+1}$ selected bunches and every bunch consists of 2^α strings, then the number of selected bunches is not greater than

$$\frac{2^\gamma \cdot 2^{\beta-\alpha+1}}{2^\alpha} = 2^{\beta+\gamma-2\alpha+1} = 2^{\gamma-\alpha+(\beta-\alpha)+O(1)} < 2^{\gamma-\alpha+\epsilon}.$$

Thus, it remains to bound the number of selected bunches that cover one string x. Assume for the sake of contradiction that there are $N = 2^{\beta-\alpha+1}$ selected bunches U_i that contain some x. By the definition of a bunch, for each of these U_i

$$U_i \subset \{y \mid \mathrm{C}(y \mid x) < \beta\}.$$

Thus, on the one hand, we have

$$\#\Big(\bigcup U_i\Big) \leq \#\Big(\{y \mid \mathrm{C}(y \mid x) < \beta\}\Big) < 2^\beta.$$

On the other hand, from the inclusion–exclusion principle, it follows that:

$$\#\Big(\bigcup U_i\Big) \geq \sum_i \#U_i - \sum_{i<j} \#(U_i \cap U_j).$$

Since $\#U_i = 2^\alpha$ and $\#(U_i \cap U_j) \leq 2^{\alpha-\Delta}$, we have

$$\#\Big(\bigcup U_i\Big) \geq N \cdot 2^\alpha - N^2 \cdot 2^{\alpha-\Delta} = 2^\beta,$$

and we get a contradiction. □

Definition 8. *We denote by* $\mathrm{Clones}_\kappa(z \mid x, y)$ *the set of "clones" of z from the perspective of (x, y) with inaccuracy $\kappa \log n$: this is the set of strings z' such that all Kolmogorov complexities involving (x, y, z') are*

(a) not greater than the corresponding quantities for (x,y,z)*, e.g.,* $\mathrm{C}(z') \le \mathrm{C}(z)$, $\mathrm{C}(z' \mid x) \le \mathrm{C}(z \mid x)$, $\mathrm{C}(x,y \mid z') \le \mathrm{C}(x,y \mid z)$*, etc.,*

(b) differ from the corresponding quantities for (x,y,z) *by at most* $\kappa \log n$*, e.g.,* $\mathrm{C}(z') \ge \mathrm{C}(z) - \kappa \log n$, $\mathrm{C}(z' \mid x) \ge \mathrm{C}(z \mid x) - \kappa \log n$, $\mathrm{C}(x,y \mid z') \ge \mathrm{C}(x,y \mid z) - \kappa \log n$*, etc.*

It can be shown by a counting argument (see [j17, j9]) that for large enough constant κ *(that does not depend on* n*) the size of* $\mathrm{Clones}_\kappa(z \mid x,y)$ *is equal to* $2^{\mathrm{C}(z|x,y)-O(\log n)}$*. We fix such a* κ *and in what follows omit a subscript in the notation* $\mathrm{Clones}(z \mid x,y)$*.*

Remark 14. A similar idea of "clones" was used on p. 59, when we extracted the common information from a pair of stochastic strings.

The next lemma is a crucial part of the proof of Theorem 26; this is the point where we use a non-Shannon type information inequality.

Lemma 5. *The set* $\mathrm{Clones}(z \mid x,y)$ *is a bunch with parameters*

$$(\mathrm{C}(z \mid x,y) - O(\log n), \mathrm{C}(z \mid x,y) - O(\log n), \mathrm{C}(z)).$$

Proof. By definition of $\mathrm{Clones}(z \mid x,y)$ we know that every element on this set has Kolmogorov complexity $\approx \mathrm{C}(z)$, and the number of elements in $\mathrm{Clones}(z \mid x,y)$ is $2^{\mathrm{C}(z|x,y)-O(\log n)}$. It remains to establish an upper bound for $\mathrm{C}(a \mid b)$ for different elements $a,b \in \mathrm{Clones}(z \mid x,y)$. To this end it is enough to prove a lower bound for $I(a:b)$.

Let a and b be elements of $\mathrm{Clones}(z \mid x,y)$. We apply inequality (1.23). By definition of $\mathrm{Clones}(z \mid x,y)$ the quantities $I(x:y \mid a)$ and $I(x:y \mid b)$ rewrite to $I(x:y \mid z)$. Since the quantities $I(x:y \mid z)$, $I(x:z \mid y)$, and $I(y:z \mid x)$ are bounded by $O(\log n)$, inequality (1.23) rewrites to

$$I(x:y) \le^+ I(a:b),$$

which implies the required upper bound for $\mathrm{C}(a \mid b)$. □

Proof of Theorem 26. We combine Proposition 3 and Lemma 5 and conclude that there exists a string w (the "core" of the bunch $\mathrm{Clones}(z \mid x,y)$) such that

$$\mathrm{C}(w) \le^+ I(x:y:z), \tag{1.27}$$

and for every $a \in \mathrm{Clones}(z \mid x,y)$

$$\mathrm{C}(a \mid w) \le^+ \mathrm{C}(z \mid x,y).$$

Claim 1: $I(w:x) \ge^+ I(x:y:z)$.

To prove this claim we observe that every a from $\mathrm{Clones}(z \mid x,y)$ forms a two-step chain

$$w \to a \to x$$

such that $\mathrm{C}(a \mid w) \le^+ \mathrm{C}(z \mid x,y)$ and $\mathrm{C}(x \mid a) \le^+ \mathrm{C}(x \mid z)$. The number of strings x that are destinations of so many chains starting at w is at most

$$\frac{2^{\mathrm{C}(z|x,y)} \cdot 2^{\mathrm{C}(x|z)}}{\#\Big(\mathrm{Clones}(z \mid x,y)\Big)} = 2^{\mathrm{C}(x|z)-O(\log n)} = 2^{\mathrm{C}(x)-I(x:y:z)-O(\log n)}.$$

The set of all strings x with this property is enumerable (given w and the values of all complexities involved). To find x given w, we need to know the index of x in this enumeration. Therefore, $\mathrm{C}(x \mid w) \leq^+ \mathrm{C}(x) - I(x:y:z)$, and Claim 1 is proven.

Observe that together with (1.27) Claim 1 implies $\mathrm{C}(w) =^+ I(x:y:z)$ and $\mathrm{C}(w \mid x) =^+ 0$.

Claim 2: $I(w:y) \geq I(x:y:z)$. The proof of this claim is similar to the proof of Claim 1. Claim 2 implies also $\mathrm{C}(w \mid y) =^+ 0$.

Claim 3: $\mathrm{C}(w \mid z) =^+ 0$. This claim follows from Claim 1, Claim 2, and a version of (1.11) in the form

$$\mathrm{C}(w \mid z) \leq^+ \mathrm{C}(w \mid x) + \mathrm{C}(w \mid y) + I(x:y \mid z).$$

Combining these claims we obtain the statement of the theorem. □

Historical remarks. In [j12] we proved a somewhat stronger version of Theorem 26, which applies in case when the quantities of $I(x:y \mid z)$, $I(x:z \mid y)$, and $I(y:z \mid x)$ are much larger than $O(\log n)$. The method of "bunches" was reused in a different context in [j9].

1.11 Constraint Information Inequalities

In this section, we discuss a class of properties of Kolmogorov complexity and Shannon's entropy that can be expressed in terms of constraint (conditional) information inequalities. Before we give a formal definition, we briefly revisit a couple of constraint information inequalities that we have seen in the previous sections.

Example 1. When talking about the common information, we observed that for all strings x, y, z

$$\text{if } \mathrm{C}(z \mid x) =^+ 0 \text{ and } \mathrm{C}(z \mid y) =^+ 0 \text{ then } \mathrm{C}(z) \leq^+ I(x:y), \tag{1.28}$$

i.e., the common information is not greater than the mutual information of x and y. This property follows immediately from the conventional (non-constraint) information inequality

$$\mathrm{C}(z) \leq^+ \mathrm{C}(z \mid x) + \mathrm{C}(z \mid y) + I(x:y),$$

which can be easily proven with the standard technique, see p. 40. A similar property holds for Shannon's entropy: for all jointly distributed (X, Y, Z)

$$\text{if } H(Z \mid X) = 0 \text{ and } H(Z \mid Y) = 0, \text{ then } H(Z) \leq I(X:Y). \tag{1.29}$$

This property also follows from the corresponding non-constraint information inequality (for entropies),

$$H(Z) \leq H(Z \mid Y) + H(Z \mid Y) + I(X:Y).$$

Example 2. In Section 1.9 we proved that for all strings a, b, x, y, z

$$\begin{aligned} &\text{if } I(x:y \mid z) =^+ I(x:z \mid y) =^+ I(y:z \mid x) =^+ 0, \\ &\text{then } I(x:y) \leq^+ I(x:y \mid a) + I(x:y \mid b) + I(a:b). \end{aligned} \tag{1.30}$$

This constraint inequality also can be deduced from the non-constraint inequality

$$I(x:y) \le^+ I(x:y \mid a) + I(x:y \mid b) + I(a:b) + I(x:y \mid z) + I(x:z \mid y) + I(y:z \mid x),$$

though the latter inequality is not trivial: this is an example of a non-Shannon-type inequality whose proof requires significant efforts.

And again, a similar property holds for Shannon's entropy: for all jointly distributed (A, B, X, Y, Z)

$$\begin{aligned} &\text{if } I(X:Y \mid Z) =^+ I(X:Z \mid Y) =^+ I(Y:Z \mid X) =^+ 0, \\ &\text{then } I(X:Y) \le^+ I(X:Y \mid A) + I(X:Y \mid B) + I(A:B). \end{aligned} \tag{1.31}$$

Obviously, this constraint inequality also can be deduced from a non-constraint (non-Shannon-type) inequality for entropies (1.24). However, in Section 1.9 we discussed a simple direct proof of (1.31) that involves no non-Shannon-type inequalities.

The very fact that constraint inequalities (1.29) and (1.31) can be obtained from the corresponding non-constraint version, has an important implication. These constraint inequalities are "robust" in the following sense: if the quantities from the constraints do not vanish and are only bounded by a small ε, then the resulting inequalities remain valid with inaccuracy $O(\varepsilon)$. Thus, the inference of (1.31) from a non-constraint non-Shannon-type inequality gives more information than a simpler direct proof.

Example 1 and Example 2 above follow the same pattern: these statements are implications where the antecedent consists of several equations (for the values of Shannon's entropy or for Kolmogorov complexity), and the consequent is a linear inequality (for Shannon's entropy or for Kolmogorov respectively). By now, we know several examples of non-trivial statements of this type for Shannon's entropy. We join all these results in one theorem:

Theorem 27. *For every tuple of jointly distributed random variables*

($\mathscr{I}1$) *if* $I(X:Y \mid A) = I(X:Y) = 0$, *then* $\square_{XY,AB} \ge 0$, *see [44],*

($\mathscr{I}2$) *if* $I(X:Y \mid A) = I(Y:B \mid A) = 0$, *then* $\square_{XY,AB} \ge 0$, *see [35],*

($\mathscr{I}3$) *if* $I(X:Y|A) = HA \mid X,Y) = 0$, *then* $\square_{XY,AB} \ge 0$, *see [j6],*

($\mathscr{I}4$) *if* $I(X:B \mid A) = I(X:A \mid B) = 0$, *then* $\square_{XY,AB} + I(X:A \mid Z) + I(X:Z \mid A) \ge 0$, *see [71],*

($\mathscr{I}5$) *if* $I(Y:A \mid B) = I(A:B \mid Y) = 0$, *then* $\square_{XY,AB} + I(Y:A \mid Z) + I(A:Z \mid Y) \ge 0$, *see [71],*

($\mathscr{I}6$) *if* $I(Y:A \mid B) = I(A:B \mid Y) = 0$, *then* $\square_{XY,AB} + I(A:B \mid Z) + I(A:Z \mid B) \ge 0$, *see [71],*

with the notation $\square_{XY,AB} := I(A:B \mid X) + I(A:B \mid Y) + I(X:Y) - I(A:B)$.

This theorem (as well as Examples 1–2 above) provides instances of *constraint information inequalities* that can be defined in general as follows.

Definition 9. *Let $\alpha(\mathscr{X})$ and $\beta_1(\mathscr{X}),\dots,\beta_m(\mathscr{X})$ be linear functions on entropies of random variables $\mathscr{X} = (X_1,\dots,X_k)$,*

$$\begin{aligned}\alpha(\mathscr{X}) &= \sum_{\emptyset\neq V\subseteq\{1,\dots,k\}} \alpha_V H(X_V),\\ \beta_i(\mathscr{X}) &= \sum_{\emptyset\neq V\subseteq\{1,\dots,k\}} \beta_{i,V} H(X_V),\ i=1\dots m\end{aligned}$$

such that the implication

$$(\beta_i(\mathscr{X}) = 0 \text{ for all } i = 1,\dots,m) \Rightarrow \alpha(\mathscr{X}) \geq 0$$

holds for all distributions $\mathscr{X}$. We call this implication a constraint linear information inequality.

Comparing Theorem 27 with Example 1 and Example 2 above, we can ask the following natural questions:

Q1. Can we reduce the constraint inequalities ($\mathscr{I}1$)–($\mathscr{I}6$) to any non-constraint information inequalities?

Q2. Are these constraint inequalities "robust," i.e., if the quantities from the constraints do not vanish and are only bounded by a small enough ε, do the resulting inequalities remain valid with a small inaccuracy?

Q3. Are there any similar constraint inequalities valid for Kolmogorov complexity?

Note that a formal definition of a constraint inequality for Kolmogorov complexity is somewhat subtle since we have to address the unavoidable "inaccuracy" of all constraints (in algorithmic information theory the value of the mutual information can be small, but we never can say that it vanishes). We discuss this issue later, and for now, focus on the Shannon's entropy framework.

We start with the negative answer to the question **Q1** above.

Definition 10. *Let $\alpha(\mathscr{X})$ and $\beta_1(\mathscr{X}),\dots,\beta_m(\mathscr{X})$ be linear functions on entropies of $\mathscr{X} = (X_1,\dots,X_n)$, and*

$$(\beta_i(\mathscr{X}) = 0 \text{ for all } i = 1,\dots,m) \Rightarrow \alpha(\mathscr{X}) \geq 0$$

be a constraint information inequality. We call this implication an essentially constraint (essentially conditional) linear information inequality, *if for all real numbers $(\lambda_i)_{1\leq i\leq m}$ the inequality*

$$\alpha(\mathscr{X}) + \sum_{i=1}^{m} \lambda_i \beta_i(\mathscr{X}) \geq 0 \tag{1.32}$$

does not hold, i.e., there exists a distribution $\mathscr{X}$ that contradicts (1.32).

The following theorem was the main result of the PhD thesis of Tarik Kaced:

Theorem 28 ([j6])**.** *Constraint inequalities ($\mathscr{I}1$)–($\mathscr{I}6$) cannot be obtained directly from any (known or yet unknown) unconditional information inequality, i.e., they are essentially constraint inequalities in the sense of Definition 10.*

Theorem 28 suggests that constraint inequalities ($\mathscr{I}1$)–($\mathscr{I}6$) are quite remarkable properties of Shannon's entropy. Do they correspond to any intuitive properties of the involved information quantities, or they are just artifacts of the definition? How sensitive are these inequalities to minor errors in the constraints? Are there any counterparts of these inequalities for Kolmogorov's or Hartley's versions of information theory? To address these (slightly informal) questions, we start with a more formal question **Q2** stated above.

Digression: geometry of entropic points. First of all, we revisit the definition of a linear information inequality and discuss the difference between entropic and almost-entropic points.

When given a k-tuple of jointly distributed random variables $\mathscr{X} = (X_1, \ldots, X_k)$, we assign to each non-empty subset random variables X_V its Shannon's entropy $H(X_V)$. We define the *entropy profile* of $\mathscr{X}$ as the vector of entropies

$$(H(X_V))_{\emptyset \neq V \subseteq \{1,\ldots,k\}},$$

let us say, in the lexicographic order. A point in $\mathbb{R}^{2^k-1}$ is called *entropic* if it is the entropy profile of some distribution $\mathscr{X}$. Thus, the conventional linear information inequalities (without constraints) are by definition the linear forms

$$\alpha(\mathscr{X}) \quad = \quad \sum_{\emptyset \neq V \subseteq \{1,\ldots,k\}} \alpha_V H(X_V)$$

that are non-negative for all entropic points.

A point is called *almost-entropic* if it belongs to the closure of the set of entropic points. It is known that for every k the set of all almost entropic points (for k-tuple of jointly distributed random variables) is a convex cone in $\mathbb{R}^{2^k-1}$, and the conventional linear information inequalities correspond to supporting half-spaces for this cone. It can be shown that for all $k \geq 3$ there exist almost-entropic but not entropic points (they all belong to the surface of this cone), though the geometric structure of these particular points remains not well understood. We refer the reader to [87] for a more technical discussion of properties of entropic and almost-entropic points.

It is not hard to see that the cone of almost-entropic points and the set of all valid linear inequalities for entropies are dual to each other. The constraint and essentially constraint inequalities also have a clear geometric interpretation in terms of the cone of almost-entropic points. However, this interpretation is hard to visualize since all nontrivial constraint inequalities belong to a high-dimensional space. Indeed, the simplest example of an essentially constraint inequality involves entropies of 4 random variables, and therefore it is an inequality in $\mathbb{R}^{15}$. To explain the geometric intuition behind the constraint information inequalities, we simplify the situation. First, we illustrate the meaning of the constraint inequalities by simplistic 2-dimensional pictures instead of 15-dimensional spaces. Second, we draw

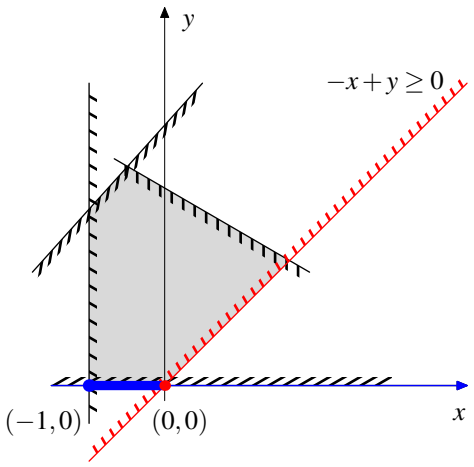

Figure 1.20: If $y=0$ then $x\leq 0$. This conditional inequality follows from $-x+y\geq 0$.

pictures not with a convex cone but with a convex geometrical set in the plane, e.g., with a convex polygon. (It is only crucial to keep the property of convexity.) Thus, the pictures below are not literal representations of information inequalities; they should be understood as metaphorical illustrations.

We start with an illustration of the geometric meaning of a non-essentially constraint inequality. Assume that a constraint inequality

$$(\beta_i(\mathscr{X})=0 \text{ for all } i=1,\dots,m)\Rightarrow \alpha(\mathscr{X})\geq 0$$

is a corollary of some non-constraint inequality (1.32). In a typical case the linear constraints $\beta_i(\mathscr{X})=0$ define some facet of the cone of almost entropic points, and the constraint inequality defines a linear boundary inside this facet. Such a boundary inside a facet of the cone results from "traces" of other non-constraint information inequalities (which correspond to other supporting hyperplanes of the cone). Geometrically, this means that the given constraint inequality can be extended to a hyperplane which provides a boundary to the entire cone. We show this effect in a simplified 2-dimensional picture. In Fig. 1.20 we are given a closed convex polygon in the plane, whose boundary consists of 5 lines. In other words, this polygon can be represented as an intersection of 5 half-spaces (defined by 5 non-constraint linear inequalities). Now we focus on the line $y=0$ and see that a point with coordinates $(x,0)$ belong to the polygon, if and only if $-1\leq x\leq 0$. This property can be stated as a constraint inequality for this polygon:

$$\text{if } y=0 \text{ then } x\leq 0. \tag{1.33}$$

Where does the bound $x\leq 0$ come from? In this example, the answer is obvious: the point $(0,0)$ is the "trace" in the line $y=0$ obtained from the non-constraint inequality $-x+y\geq 0$. In other words, constraint inequality (1.33) can be extended to the non-constraint inequality $-x+y\geq 0$.

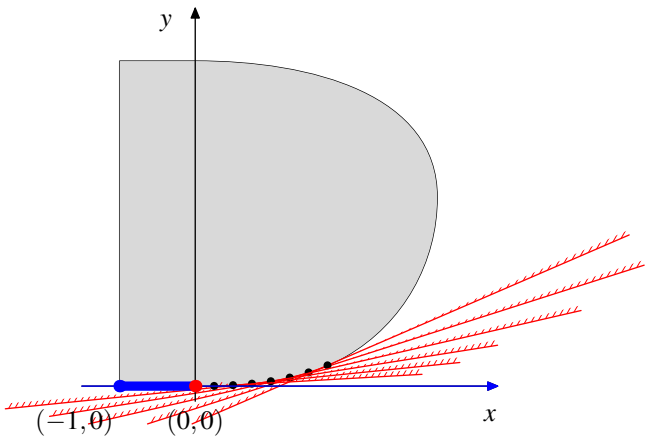

Figure 1.21: If $y = 0$ then $x \le 0$. This conditional inequality is implied by an infinite family of tangent half-planes.

We proceed with a more involved example shown in Fig. 1.11. In this picture, the geometrical figure (a closed convex area shown in gray) is not a polygon: its boundary contains a smooth curve. We assume that the tangent line to this curve at the point $(0,0)$ is horizontal. Similarly to the previous example, we focus on the line $y = 0$. The point with coordinates $(x,0)$ belongs to the gray figure if and only if $-1 \le x \le 0$. That is, constraint inequality (1.33) applies to this convex figure. However, in this case, the constraint inequality cannot be extended to any unconditional linear inequality: whatever λ is given, the inequality $-x + \lambda y \ge 0$ is not valid for the gray area in the picture. In other words, here we have an example of an essentially constraint inequality.

In some sense, in this example, the inequality $-1 \le x \le 0$ follows from an infinite family of non-constraint inequalities. Indeed, (1.33) is implied by the infinite family of supporting half-planes that are tangent to the gray area in the picture. This phenomenon can occur for a convex closed body only if this body is not polyhedral (respectively, not polygonal in the 2-dimensional case). A formal proof of this fact involves the usual machinery of the convex analysis (Farkas' lemma), see [j6] for details.

Let us proceed with another example. In our last example (Fig. 1.22), the gray area is still a convex set, but now it is not closed: we assume that the part of its boundary shown by the bold lines belongs to the figure, and the part of the boundary shown by the dashed lines does not. The same constraint inequality (1.33) applies to this area. Obviously, this inequality is essentially constraint, i.e., it cannot be extended to $-x + \lambda y \ge 0$ with any λ. Moreover, (1.33) holds for the gray area but it does not hold for its closure. For example, the point $(1,0)$ does not satisfy this conditional inequality.

The three examples discussed above help us to understand the geometry of the constraint information inequalities. Each constraint linear inequality for Shannon's

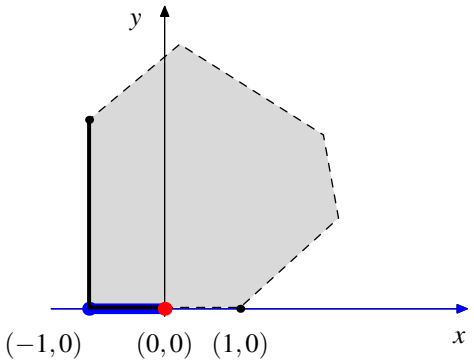

Figure 1.22: If $y = 0$ then $x \le 0$. This conditional inequality does not hold for the *closure* of the grey area.

entropies is a linear inequality that is valid on some facet on the set of all entropic points. There are three substantially different types of constraint information inequalities.

> **Case 1.** If an inequality is not essentially constraint, then it can be extended to a conventional (unconditional, non-constraint) inequality, which corresponds to a supporting half-space for the whole cone of almost entropic points. This case corresponds to the illustration in Fig. 1.20.
>
> **Case 2.** For an essentially constraint inequality, there are two different possibilities:
>
> > **Case 2(a).** If a constraint inequality is valid not only for entropic but also for almost entropic points, then this inequality follows from an infinite family of non-constraint inequalities, similarly to the example in Fig. 1.21. From the very fact that such conditional information inequalities exist it follows that the cone of almost entropic points (for $n \ge 4$ random variables) is not polyhedral. In other words, every essentially constraint inequality of this type provides an alternative proof of the Matúš theorem from [71], see [j6, Section 6] for a more detailed discussion.
> >
> > **Case 2(b).** If a constraint inequality does not apply to almost entropic points, then its geometric meaning is similar to the illustration in Fig. 1.22.

Now, having completed the geometric digression, we proceed with a more formal classification of the known constraint information inequalities.

Theorem 29. *(a) [71] Inequalities ($\mathscr{I}4$)–($\mathscr{I}6$) are valid for almost entropic points* (*see Case 2(a) above*).

(b) [j6] Inequalities ($\mathscr{I}1$) and ($\mathscr{I}3$) do not hold for almost entropic points (*see Case 2(b) above*).

Remark 15. It remains unknown whether inequality ($\mathscr{I}2$) holds for almost entropic point (whether it belongs to the Case 2(a) or to the Case 2(b) discussed above).

Remark 16. The essentially constraint inequalities from the Case 2(a) are "robust" in the following weak sense: for every $\delta > 0$ there exists an $\varepsilon > 0$ such that if the quantities from the constraint are bounded by ε, then the resulting inequality is valid with inaccuracy δ. However, ε goes to zero faster than a linear function of δ.

The essentially constraint inequalities from the Case 2(b) are non-robust in a very strong sense: even a minor violation of the constraints leads to inaccuracy $\Omega(1)$ in the resulting inequality.

It remains to address the question **Q3** from p. 75 and discuss counterparts of the known essentially constraint information inequalities in the framework of algorithmic information theory. Not surprisingly, the situation is pretty different for the inequalities from Cases 2(a) and Case 2(b). For all known inequalities of the first type (those which are valid for almost-entropic points, Cases 2(a)) we can prove a version with Kolmogorov complexity:

Theorem 30 ([j6])**.** *Let $f(n)$ be a function of an integer argument such that $f(n) \geq \log n$ for all n. Then there exists a constant $\kappa > 0$ such that for every tuple of binary strings (a, b, c, d, e)*

($\mathscr{I}4$-Kolm) if $I(a : d \mid c) \leq f(n)$ and $I(a : c \mid d) \leq f(n)$, then

$$\square_{ab,cd} + I(a : c \mid e) + I(a : e \mid c) + \kappa \cdot \sqrt{n \cdot f(n)} \geq 0,$$

($\mathscr{I}5$-Kolm) if $I(b : c \mid d) \leq f(n)$ and $I(c : d \mid b) \leq f(n)$, then

$$\square_{ab,cd} + I(b : c \mid e) + I(c : e \mid b) + \kappa \cdot \sqrt{n \cdot f(n)} \geq 0,$$

($\mathscr{I}6$-Kolm) if $I(b : c \mid d) \leq f(n)$ and $I(c : d \mid b) \leq f(n)$, then

$$\square_{ab,cd} + I(c : d \mid e) + I(c : e \mid d) + \kappa \cdot \sqrt{n \cdot f(n)} \geq 0,$$

where n denotes $\mathrm{C}(a, b, c, d, e)$, and $\square_{ab,cd} := I(c : d \mid a) + I(c : d \mid b) + I(a : b) - I(c : d)$.

In this theorem $f(n)$ measures inaccuracy of the constraints. For example, assuming $I(a : d \mid c) = O\left(\sqrt{n}\right)$ and $I(a : c \mid d) = O\left(\sqrt{n}\right)$ we conclude that there exists a $\theta > 0$ such that for all a, b, c, d, e

$$\square_{ab,cd} + I(a : c \mid e) + I(a : e \mid c) + \theta \cdot n^{3/4} \geq 0.$$

Theorem 30 provides a trade-off between two measures of inaccuracy: $f(n)$ measures inaccuracy of the constraints, and $O(\sqrt{n \cdot f(n)})$ is a bound for inaccuracy of the conclusions. It can be shown that the suggested trade-off is tight. In particular, given the constraints with inaccuracy $O(\log n)$ we cannot get the resulting inequality with any better precision than $O(\sqrt{n \log n})$, see [j6] for details.

The next theorem shows that in some sense counterparts of ($\mathscr{I}1$) and ($\mathscr{I}3$) do not hold for Kolmogorov complexity. We argue that even a minor inaccuracy $O(\log n)$ in the constraints results in the definitive failure of the implied linear inequality:

Theorem 31 ([j6]). *(a) There exists an infinite sequence of tuples of strings (a,b,c,d) such that the lengths of all strings a,b,c,d are $\Theta(n)$, $I(a:b)=O(\log n)$, $I(a:b\mid c)=O(\log n)$, and*

$$I(c:d)-I(c:d\mid a)-I(c:d\mid b)=\Omega(n)$$

(b) There exists an infinite sequence of tuples of strings $(a,b,c,d)_n$ such that the lengths of all strings a,b,c,d are $\Theta(n)$, $C(c\mid a,b)=O(\log n)$, $I(a:b\mid c)=O(\log n)$, and

$$I(c:d)-I(c:d\mid a)-I(c:d|b)-I(a:b)=\Omega(n).$$

On the other hand, there is one positive result: some version of ($\mathscr{I}3$) is valid for Kolmogorov complexity if we impose the constraints in a very strict form. We discuss this constraint inequality in Section 1.14.

Historical remark: The first example of a non-trivial constraint inequality for Shannon's entropy was discovered in [41]. In a more general context constraint information inequalities were discussed in [87]. Constraint inequalities were used (somewhat implicitly) in [71] in a proof of an infinite family of information inequalities. The technical definition of an essentially constraint information inequality was suggested in [c17]. Most results of this section appeared in [j6] (see the Appendix to this manuscript). This research (including purely probabilistic results) was motivated by the interplay between Shannon's and Kolmogorov's versions of information theory: we were trying to translate the known constraint inequalities for Shannon's entropy in the language of Kolmogorov complexity. As we have shown above, in some cases this translation succeeds, and in others, it fails.

1.12 Combinatorial Interpretation of the Conventional Information Inequalities

In this section, we discuss the parallelism between information inequalities in the combinatorial, probabilistic, and algorithmic frameworks, with a focus on the combinatorial version. We start the discussion with a few examples.

Example 1. We start with arguably the simplest example of information inequality,

$$\mathrm{C}(x,y)\le^{+}\mathrm{C}(x)+\mathrm{C}(y). \tag{1.34}$$

Intuitively, it claims that the shortest description of a pair (x,y) is not longer than the sum of the lengths of the shortest descriptions of x and y. The proof is trivial: we can join the descriptions of x and y and obtain a joint description of a pair (possibly, not optimal). Only a logarithmic overhead is needed to encode the coupling of two programs in one.

We keep in mind the basic idea of Hartley's approach: we gauge the information that is necessary to encode an arbitrary element of a given finite set, so the natural measure of this information is the (binary) logarithm of a cardinality of a set. Let S be a finite set of pairs of binary strings. Than Hartley's measure of information in this set is $\log\#S$. If we want to measure separately the information contained in

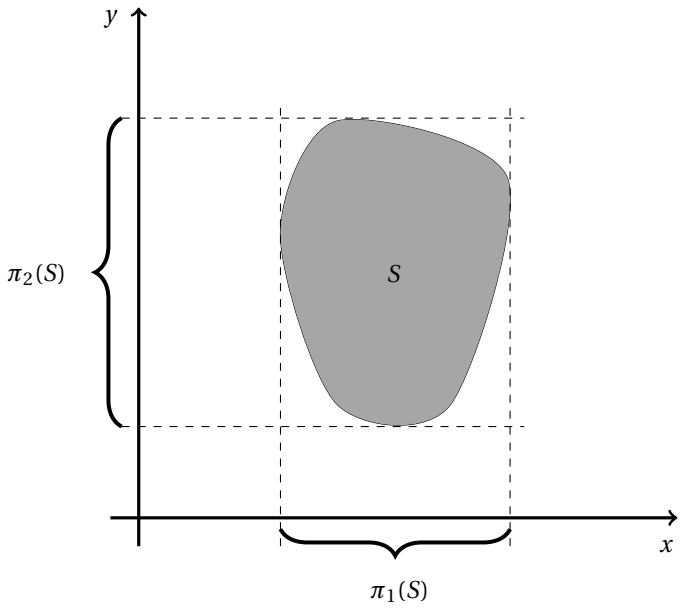

Figure 1.23: The cardinality of a set S is not greater than the product of the cardinalities of its projections, $\pi_1(S)$ and $\pi_2(S)$.

both components of these pairs, we should consider the projections of S on both coordinates,

$$\begin{aligned} \pi_1(S) &:= \{x : \text{there is a } y \text{ such that } (x,y) \in S\}, \\ \pi_2(S) &:= \{y : \text{there is an } x \text{ such that } (x,y) \in S\}, \end{aligned}$$

and take the logarithms of each of these projections. Then we observe that a joint encoding of pairs in S requires not more bits than the sum of bits in separate encoding of elements in $\pi_1(S)$ and $\pi_2(S)$. The trivial statement "Hartley's information in a pair is not greater than the sum of Hartley's information in both components" can be written as

$$\log \#S \le \log \#\pi_1(S) + \log \#\pi_2(S) \tag{1.35}$$

or, without logarithms, as

$$\#S \le \#\pi_1(S) \cdot \#\pi_2(S). \tag{1.36}$$

Inequality (1.36) is self-evident: it claims that the cardinality of a set is not greater than the product of the cardinalities of its projections, Fig. 1.23. However, in this toy example, we observe the parallels between (1.34) and (1.36). This example suggests a general scheme how an inequality for Kolmogorov complexity can be translated in a combinatorial statement.

Example 2. Let us consider a slightly stronger version of (1.34),

$$\mathrm{C}(x,y) \le^+ \mathrm{C}(x) + \mathrm{C}(y \mid x). \tag{1.37}$$

This inequality is also very intuitive: the shortest description of a pair (x, y) is not longer than the sum of the lengths of the shortest description of x and the shortest description for y given x as a condition. Let us find a natural combinatorial version of this inequality. Let S again be a finite set of pairs of binary strings. We suppose again that the natural counterparts of $C(x, y)$ and $C(x)$ are $\log\#S$ and $\log\#\pi_1(S)$ respectively. It remains to define a natural combinatorial version of the conditional complexity $C(y \mid x)$.

Since Hartley's idea was to measure the information "in the worst case," we suggest to take the quantity

$$\log \max_{x \in \pi_1(S)} \#S_x,$$

where S_x is a "section" of the set,

$$S_x := \{y : (x, y) \in S\}.$$

In other words, we take the number of digits required to specify all pairs (x, y) in S assuming that the first component of the pair is already known. So we take the maximum over the "vertical sections" of S corresponding to all possible x.

Now a natural combinatorial rephrasing of (1.37) is

$$\log\#S \le \log\#\pi_1(S) + \log[\max_{x \in \pi_1(S)} \#S_x\,].$$

Without logarithms this inequality rewrites to

$$\#S \le \#\pi_1(S) \cdot [\max_{x \in \pi_1(S)} \#S_x\,], \tag{1.38}$$

i.e., the cardinality of S is bounded by the product of the cardinality of its first projection and the maximal size of the vertical section, see Fig. 1.24.

Inequalities (1.36) and (1.38) look pretty trivial. In the next example we apply a similar translation to a more interesting information inequality and end up with a more involved combinatorial statement.

Example 3. We proceed with a less trivial example

$$2C(x, y, z) \le^{+} C(x, y) + C(x, z) + C(y, z) \tag{1.39}$$

(see a discussion on p. 39). This is an inequality for a triple of strings, so the counterpart in the combinatorial world should be a statement about a three-dimensional set. Let S be a finite set of triples of strings. Hartley's information in the triple is defined, as usual, as the logarithm of the cardinality of S. To deal with pairs selected from a triple, we consider the projections of S onto two-coordinate spaces:

$$\begin{aligned}
\pi_{12}(S) &:= \{(x, y) : \text{there is a } z \text{ such that } (x, y, z) \in S\},\\
\pi_{13}(S) &:= \{(x, z) : \text{there is a } y \text{ such that } (x, y, z) \in S\},\\
\pi_{23}(S) &:= \{(y, z) : \text{there is an } x \text{ such that } (x, y, z) \in S\},
\end{aligned}$$

and the combinatorial versions of $C(x, y)$, $C(x, z)$, and $C(y, z)$ are the quantities

$$\log\#\pi_{12}(S),\ \log\#\pi_{13}(S),\ \text{and}\ \log\#\pi_{23}(S)$$

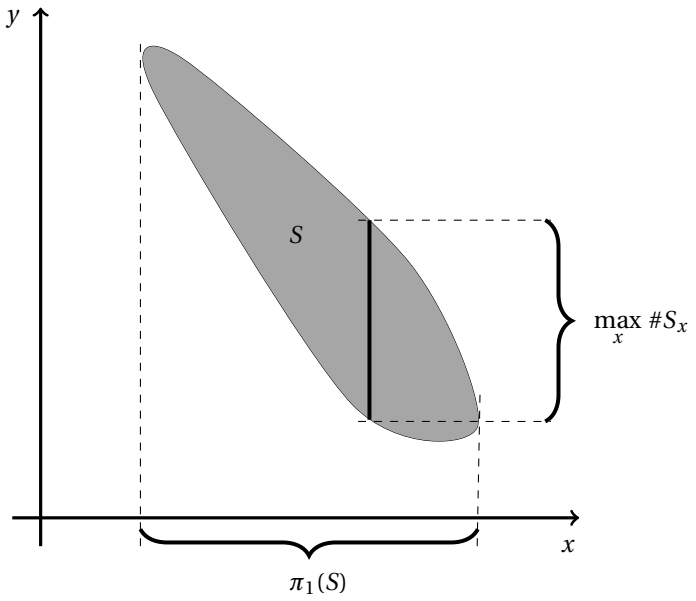

Figure 1.24: The cardinality of a set S is not greater than the product of the cardinality of its projection on the first coordinate and the maximum size of its vertical section.

respectively. Therefore, a natural combinatorial version of (1.39) is the inequality

$$2\log\#S \le \log\#\pi_{12}(S) + \log\#\pi_{13}(S) + \log\#\pi_{23}(S).$$

The information-theoretic meaning of this inequality can be worded as follows: the number of bits required to encode all triples in S is not greater than the half-sum of the number of bits required to encode separately the pairs associated to these triples. Without logarithms this property rewrites to

$$(\#S)^2 \le \#\pi_{12}(S)\cdot\#\pi_{13}(S)\cdot\#\pi_{23}(S). \tag{1.40}$$

A continuous version (1.40) would say that the volume of a three-dimensional body is not greater than the square root of the product of areas of the projections of the body onto three coordinate planes, see Fig. 1.25. Inequality (1.40) is known as Shearer's lemma (more precisely, a special case of combinatorial Shearer's lemma, [23], [33]). The conventional proof of this combinatorial statement uses Shannon's entropy. Loosely speaking, to get (1.40) we introduce the uniform distribution (X,Y,Z) on S, apply to the obtained three-dimensional distribution the inequality

$$2H(X,Y,Z) \le H(X,Y) + H(X,Z) + H(Y,Z),$$

and then observe that entropies $H(X,Y)$, $H(X,Z)$, $H(Y,Z)$ are not greater than logarithms of the cardinalities of the corresponding projections of S. An alternative proof uses Kolmogorov complexity and inequality (1.39), see [j14].

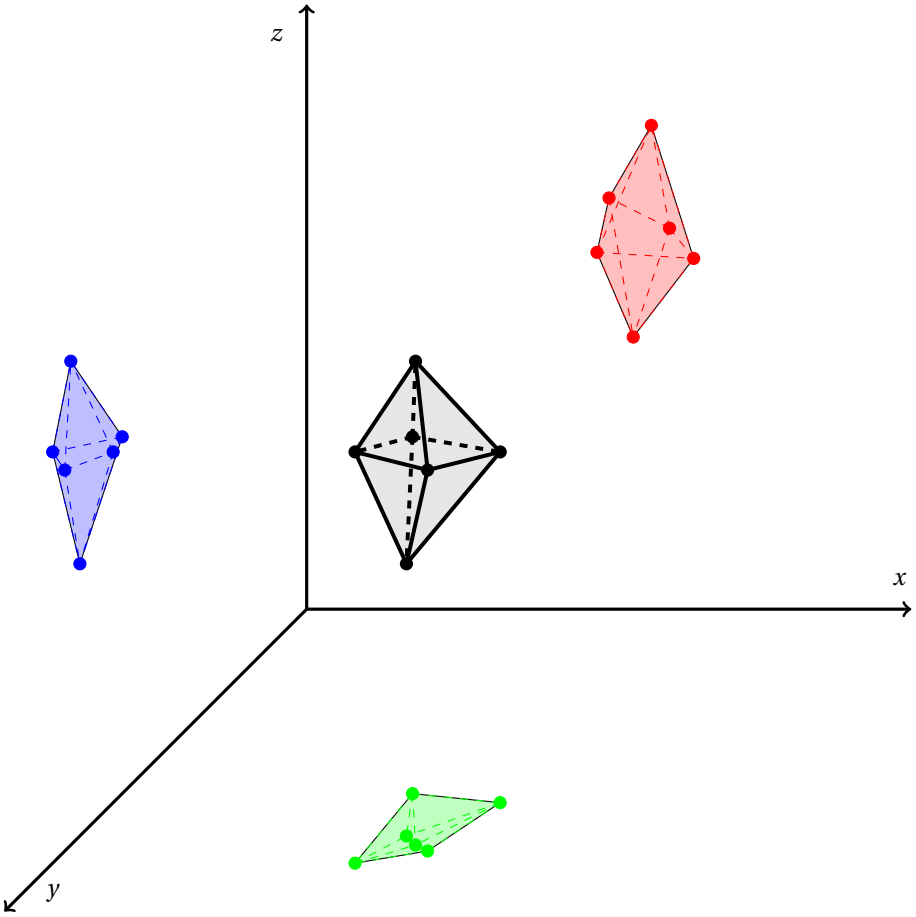

Figure 1.25: The square of the volume of a three-dimensional body (shown in gray) is not greater than the product of the areas of the body's projections on the three coordinate planes (the projections are shown in color).

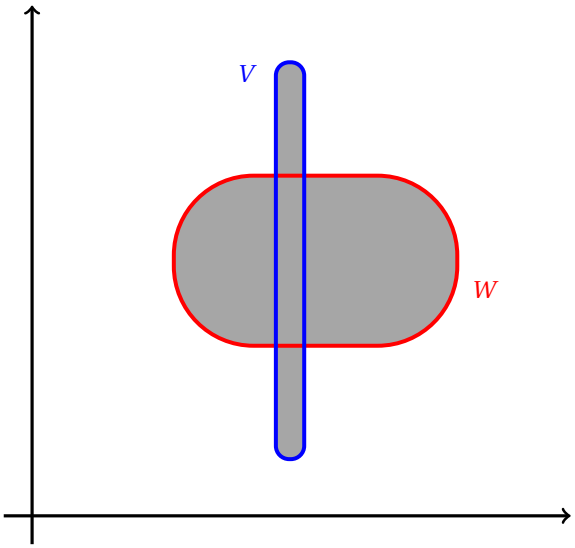

Figure 1.26: A set (shown in gray) breaks up into two parts V and W: V has long vertical sections and a small projection on the first coordinate, and W has shorter vertical sections and a large projection on the first coordinate.

Examples 1-3 suggest that a natural way of translating information inequalities in the combinatorial language provides a meaningful (and non-trivial, as we see in Example 3) application of information theory to combinatorics. Now we aim to formulate a general rule that transforms an inequality for Kolmogorov complexity in the corresponding combinatorial statement, extending the observations from Examples 1-3. In what follows we propose such a rule. But first we discuss one more simple example that shows that the general translation can be slightly more involved than Examples 1-3 discussed above.

Example 4. Now we try to transform in a combinatorial form the inequality

$$\mathrm{C}(x) + \mathrm{C}(y \mid x) \leq^{+} \mathrm{C}(x, y), \tag{1.41}$$

which is the hard direction of the Kolmogorov–Levin theorem. In Example 2 we already defined the combinatorial quantities corresponding to the complexities $\mathrm{C}(x, y)$, $\mathrm{C}(x)$, and $\mathrm{C}(y \mid x)$. A naive translation of (1.41) looks as follows: for every set of pairs S,

$$\#\pi_1(S) \cdot [\max_{x \in \pi_1(S)} \#S_x] \leq \#S. \tag{1.42}$$

However, this statement is obviously false. See, for example, the set S in Fig. 1.26, which has large S_x for a few particular x and small S_x for many other x.

In fact, Fig. 1.26 suggests a more plausible statement which looks parallel to (1.41). We cannot say that S has at once a small projection and small vertical

sections. But we can split S into two parts so that the first one has a small projection and the second one has small vertical sections. Here is the technical statement:

> Let v and w be two integers such that $vw \geq \#S$. Then S can be partitioned into $S = V \cup W$ with $\#\pi_1(V) \leq v$ and $\max_x \#W_x \leq w$. (1.43)

It can be shown that (1.43) is indeed true for all S, see [j14].

Now we are ready to present the main result of this section. We propose a general rule that translates a linear information inequality in an equivalent combinatorial statement (the initial inequality holds for the Kolmogorov complexity of all tuples of strings if and only if the resulting combinatorial statement is true for all finite sets). We apply this translation to all linear inequalities with the quantities of conditional and unconditional Kolmogorov complexity. Given a linear inequality, we separate the terms so that all coefficients on the left-hand side and on the right-hand side are positive. Thus, every inequality can be represented as

$$\sum_{(I,J)\in\mathscr{A}} \alpha_{I,J}\mathrm{C}(x_I \mid x_J) \leq^+ \sum_{(I,J)\in\mathscr{B}} \beta_{I,J}\mathrm{C}(x_I \mid x_J), \tag{1.44}$$

where all $\alpha_{I,J}$ and $\beta_{I,J}$ are positive real coefficients and $\mathscr{A},\mathscr{B}$ are disjoint sets of pairs of disjoint subsets of $\{1,\dots,k\}$. (We assume that I in all pairs is non-empty; if J is empty, the value $\mathrm{C}(x_I|x_J)$ is understood as the unconditional complexity $\mathrm{C}(x_I)$.)

We suppose to give a combinatorial statement that is equivalent to (1.44). To this end we need a general version of the "combinatorial entropy" and the "combinatorial conditional entropy," similar to the quantities used in Example 1–4 above. We define them as follows. Let S be a set of k-tuples, and let I and J be disjoint subsets of the indices in $\{1,\dots,k\}$. For any element $a \in S$ we consider the section of S going through a obtained when all J-coordinates are fixed; consider the projection of this section onto the coordinates from I. The cardinality of this projection depends on a; we denote by $n_{I|J}(S)$ the maximal cardinality of this projection (maximum over all a in S). In other words, we fix the J-coordinates of a point $a \in S$ and consider the set of all possible values of the I-coordinates, and the maximum cardinality of this set is denoted by $n_{I|J}(S)$. Observe that the notation $n_{I|J}(S)$ makes sense for the empty set of indices J (e.g., $n_{\{1,\dots,k\}|\emptyset}(S)$ denotes the cardinality of the entire S). Given this notation, we can translate (1.44) in the equivalent combinatorial statement.

Theorem 32 ([j14])**.** *Inequality* (1.44) *is valid for given coefficients $\alpha_{I,J}$ and $\beta_{I,J}$ and for any strings $x_1,\dots,x_k$ (up to a logarithmic term) if and only if the following combinatorial statement is true:*

there exists a polynomial $\mathrm{poly}(n)$ *such that for any n, for any set $S \subset (\{0,1\}^n)^k$ and for any integers $\gamma_{I,J}$ such that*

$$\prod_{(I,J)\in\mathscr{B}} [n_{I|J}(S)]^{\beta_{I,J}} \leq \prod_{(I,J)\in\mathscr{A}} \gamma_{I,J}^{\alpha I,J} \tag{1.45}$$

the set S can be covered by sets $U_{I,J}$ (for $(I,J) \in \mathscr{A}$) such that

$$n_{I|J}(U_{I,J}) \leq \gamma_{I,J} \cdot \mathrm{poly}(n).$$

We interpret this theorem as a tool to apply information theory to combinatorial problems. Theorem 1.45 seems to be more complicated than one could expect looking at Examples 1-4 above: it involves a polynomial factor that corresponds to an additive logarithmic term in the inequalities for Kolmogorov complexity. In [j14] we showed that a simpler version of the combinatorial statement (without the poly(n) term) can be proven for a large class of information inequalities. However, it remains open whether this term can be omitted in the general version of Theorem 1.45. Another combinatorial interpretation of information inequalities was suggested in [69] (it involves a version of (1.45) with a tighter bound but requires to split the given set in poly(n) "quasi-uniform" parts).

1.13 Towards a Combinatorial Interpretation of Constraint Information Inequalities

By now, we do not know any general scheme of translation of the constraint information inequalities into equivalent combinatorial statements. We keep in mind such a translation as an eventual goal and discuss several particular combinatorial applications of constraint information inequality.

We start with recalling inequality ($\mathscr{I}3$) for Shannon's entropy:

$$I(X:Y \mid A) = 0 \text{ and } H(A \mid X,Y) = 0 \tag{1.46}$$

$$\Downarrow$$

$$I(A:B) \le I(A:B \mid X) + I(A:B \mid Y) + I(X:Y) \tag{1.47}$$

(see Theorem 27 on p. 74). Observe that under the assumption (1.46) the concluding inequality (1.47) can be equivalently rewritten as

$$H(A \mid B,X) + H(A \mid B,Y) \le H(A \mid B). \tag{1.48}$$

Thus, ($\mathscr{I}3$) can reformulated as "(1.46) $\Rightarrow$ (1.48)."

Now we are going to make the constraint (1.46) weaker (and, therefore, make the entire statement stronger). More specifically, we consider the following constraint imposed on a distribution (a,x,y):

$$\begin{aligned}&\text{For each quadruple of values } a,a',x,y,\\ &\text{if the probabilities of all four events}\\ &[A=a,X=x],\ [A=a,Y=y],\\ &[A=a',X=x],\ [A=a',Y=y]\\ &\text{are positive, then } a=a'.\end{aligned} \tag{1.49}$$

It is not hard to verify (see [j2] for details) that (1.49) is indeed weaker than (1.46) (the latter implies the former but not visa-versa). The same time, the weaker version of the condition is enough to imply (1.48):

Theorem 33 ([j2])**.** *Inequality* (1.48) *is true for all distributions satisfying* (1.49).

The implication "(1.49) ⇒ (1.48)" is interesting even for distributions (A, B, X, Y) with a trivial B (where B is the random variable with one single value and zero entropy). In what follows we focus on this special case. In other words, we consider the inequality

$$H(A \mid X) + H(A \mid Y) \le H(A) \tag{1.50}$$

and the implication (1.49) ⇒ (1.50), which is a special case of Theorem 33. In fact, under the assumption (1.49) we do not lose much generality when we reduce the variable B. Indeed, the property (1.49) is purely negative: it requires that the probabilities of some events vanish. Therefore, the same property applies to all conditional distributions for (A, X, Y) conditional on every fixed values of B. Thus, the general version of (1.48) can be obtained as the average of (1.50) over all distributions conditional on the possible values of B.

Every distribution where the value of A is a deterministic function of (X, Y), can be naturally interpreted as a distribution on a bipartite graph with colored edges. We understand (X, Y) as a randomly chosen pair of adjacent vertices (X form the left part and Y from the right part of the graph), and A as the color of the edge connecting X and Y. In terms of this graph (1.49) can be understood as follows.

Definition 11. *An edge coloring of a graph is an assignment of colors to the edges of the graph so that every two adjacent edges have different colors.*

We say that an edge coloring in a bipartite graph is rich *if for every pair*

$$\langle \textit{left vertex } x,\ \textit{right vertex } y \rangle$$

there is at most one color a touching both x and y (the latter means that there is an edge with color a incident to x and an edge, maybe a different one, with color a incident to y). This property is required for all pairs of vertices (x, y), including those that are not connected by an edge.

Proposition 4. *For every bipartite graph with a rich coloring and for every distribution on the edges of this graph, the random tuple*

$$(\textit{left vertex } x,\ \textit{right vertex } y,\ \textit{color of the edge})$$

satisfies (1.50).

(This proposition is a rewording of Theorem 33 for trivial B.)

Remark 17. Keeping in mind the constraint $H(A \mid X, Y) = 0$, we can rewrite (1.50) to $I(A : X : Y) \ge 0$.

We proceed with a simple and straightforward application of this proposition:

Corollary 4. *Assume that the degree of each left vertex in a given bipartite graph is at least L and the degree of each right vertex is at least R. Then, the number of colors in every rich edge coloring of the graph is at least LR.*

Sketch of proof: Define the uniform distribution on the set of edges of the graph and denote by (A, X, Y) the random triple ⟨left vertex x, right vertex y, color of the edge⟩. As the coloring is rich, we apply Proposition 4 and obtain (1.50).

By construction, the distribution on the edges is uniform. Therefore, for each vertex x, the conditional distribution of edges incident to this x is also uniform. In a valid edge coloring all edges incident to one and the same vertex must have different colors. Hence, for every vertex x, all colors touching this x are equiprobable. Thus, conditional on $X = x$, the random variable A is uniformly distributed on the set of colors of edges compatible with x. Therefore, $H(A \mid X) \geq \log L$. A similar argument gives $H(A \mid Y) \geq \log R$. It remains to apply (1.50) and obtain $H(A) \geq \log L + \log R$. It follows that the range of A is at least LR. □

Corollary 4 can be seen as a combinatorial interpretation of the constraint inequality from Theorem 33. Before we proceed with another application of Proposition 4, we recall the following definition from graph theory:

Definition 12. *For any bipartite graph $G = (V_1, V_2, E)$ (with the set of vertices $V_1 \cup V_2$ and the set of edges $E \subset V_1 \times V_2$) its* biclique cover number $bcc(G)$ *is defined as the minimal number of bicliques (complete bipartite subgraphs) that cover all edges of G.*

Biclique coverings play an important role in communication complexity. Specifically, the *non-deterministic communication complexity* (see [40]) of a predicate

$$P : U \times U \to \{0, 1\}$$

can be defined as $\log bcc(G)$ for the bipartite graph $G = (V_1, V_2, E)$, where $V_1 = V_2 = U$, and E is the set of all pairs $(x, y) \in U \times U$ such that $P(x, y) = 1$.

Corollary 5. *Assume that the edges of a bipartite graph $G = (V_1, V_2, E)$ are colored in such a way that*

$$\begin{aligned} &\textit{if edges } (x, y') \textit{ and } (x', y) \textit{ of the graph have the same color } a, \textit{ and} \\ &\textit{vertices } x \textit{ and } y, \textit{ as well as vertices } x' \textit{ and } y', \textit{ are also connected} \\ &\textit{by edges, then the latter two edges also have color } a. \end{aligned} \tag{1.51}$$

Assume further that a probability distribution over the edges of the graph is given. Denote by (X, Y, A) the random variables where

- $X = [\textit{the left end of the edge}]$,
- $Y = [\textit{the right end of the edge}]$,
- $A = [\textit{the color of the edge}]$.

Then $bcc(G) \geq 2^{\frac{1}{2}(H(A|X)+H(A|Y)-H(A))}$.

(Corollary 5 follows pretty easily from Proposition 4, see [j2] for details.)

Example. We apply Corollary 5 to a bipartite graph which often plays in communication complexity the role of a benchmark for different methods. Consider the bipartite Kneser graph $KG_{n,k} = (V_1, V_2, E)$, where both parts V_1 and V_2 consist of all

k-element subsets of $\{1,\dots,n\}$, and the set of edges $E \subset V_1 \times V_2$ consists of all pairs of disjoint sets. We want to estimate the biclique cover number for this graph. This question corresponds to the following communication problem. Consider a conversation between two parties (Alice and Bob), assuming that each party is given a k-element subset in $\{1,\dots,n\}$. Alice and Bob want to check whether their two subsets have an empty intersection. The question is how many bits of information they should send to each other to make sure that their inputs are indeed disjoint. For deterministic communication protocols, the value $\log bcc(KG_{n,k})$ provides a lower bound on the communication complexity of the problem. For non-deterministic protocols, this value essentially coincides with the communication complexity.

To apply Corollary 5, we should define a suitable coloring on the edges of the graph. We assign to each edge (x,y) the color $x \cup y$ (the union of the sets x and y) and consider the uniform probability distribution on the edges of this graph.

Condition (1.51) is satisfied. Indeed, assume we are given three pairs of disjoint k-element subsets: (x,y), (x,y') and (x',y), and assume also that $x \cup y' = x' \cup y = a$. It follows that $x = x'$ and $y = y'$, and therefore $x \cup y = a$.

Thus, we can apply Corollary 5:

$$bcc(KG_{n,k}) \geq 2^{\frac{1}{2}[H(A|X)+H(A|Y)-H(A)]}. \tag{1.52}$$

There are $\binom{n}{2k}$ equiprobable colors of edges, and therefore $H(A) = \log_2 \binom{n}{2k}$. On the other hand, $H(A \mid X) = H(A \mid Y) = \log_2 \binom{n-k}{k}$. Inequality (1.52) rewrites to

$$bcc(KG_{n,k}) \geq \sqrt{\binom{n-k}{k}^2 \Big/ \binom{n}{2k}}.$$

For $n \gg k$ this bound implies $bcc(KG_{n,k}) \geq 2^{\Omega(k)}$. On the other hand, it is known that $bcc(KG_{n,k}) \leq 2^{O(k+\log\log n)}$ (see [40, Section 2.3]). Thus, in the case $\Omega(\log\log n) \leq k \ll n$ our lower bound is almost tight.

The achieved bound is by no means new: the standard fooling set technique (see [40]) proves for this graph the bound $bcc(KG_{n,k}) \geq \binom{2k}{k}$ for all $n \geq 2k$. However, this simple example illustrates the connection between biclique cover and conditional information inequalities. We presume that communication complexity is a natural area of application for constraint information inequalities. In the next section we use a similar argument to prove a bound for the protocols of common secret key agreement (in a situation where the other known techniques seem to be less efficient).

Historical remarks. The questions on combinatorial and algorithmic interpretation of essentially constrain inequalities motivated several recent publications (e.g., [c17, j6, j2]). We believe that clique covering and similar properties of graphs provide a suitable language for interpretation of inequalities like ($\mathscr{I}1$) or ($\mathscr{I}3$). However, a complete and consistent explanation of the phenomenon of essentially constraint inequalities remains elusive. We refer the reader to [j2] (see Appendix to the manuscript) for some more examples of combinatorial applications of constraint

information inequalities and for technical proofs of the results presented in this section.

1.14 Common Secret Key Agreement

In this section, we discuss an interpretation of mutual information from the cryptographic perspective and give a sort of operational characterization of mutual information in the framework of Kolmogorov complexity. More specifically, we show that the mutual information of a pair of binary strings x, y is essentially the maximal size of a shared secret key that two parties (one of them is given x as an input, and the other one is given y) can obtain by using a probabilistic communication protocol via a public channel. We start with a simple example.

Example. Assume two parties (Alice and Bob) want to agree on a common secret key. Alice and Bob can communicate through a public channel. We assume also that in the very beginning Alice is given a random line x in the affine plane over the finite field with 2^n elements, and Bob is given a random point y on this line.

The line x in the affine plane can be specified by the slope a and the intercept b, and the point y in the plane can be specified by its two coordinates c and d. Therefore, both x and y contain $2n$ bits of information. Due to the geometrical correlation, together they have $3n$ bits of information.

We emphasize that x is known only to Alice (Bob merely knows that this line is incident with his point y), and y is known only to Bob (Alice merely knows that the point is incident with her line x). The adversary only knows *a priori* that the line and the point are incident with each other. All information sent via the communication channel is accessible to the adversary.

In this setting, the mutual information between x and y is n bits, so in some sense Alice and Bob share n bits of the mutual information. We argue that they can "materialize" this mutual information and obtain a common secret key of size n. They can operate as follows:

- Alice sends a to Bob,
- Bob (keeping in mind that his point is incident with the line) reconstructs x,
- Alice and Bob use b (the intercept of the line) as the secret key.

Observe that the adversary can learn the value a by intercepting the communication, but this gives no information on b.

The solution proposed in this simple example employs crucially the very specific geometrical relation between x and y. However, it can be shown that Alice and Bob can agree on a common secret key in a much more general setting. We consider secret key agreement protocols where Alice and Bob are given inputs x and y respectively. We assume that Alice and Bob also know how their x and y are correlated. More technically, we assume that Alice and Bob know the complexity profile of x and y, i.e., the values of Kolmogorov complexity $\mathrm{C}(x), \mathrm{C}(y), \mathrm{C}(x,y)$ (or at least some approximations of these values). We admit randomized protocols: Alice and

Bob can secretly toss a coin and obtain therefore their private sequences of random bits.

Theorem 34 ([e1, c1], informal statement)**.** *In the setting specified above we prove the following positive and negative statements.*

1. *There is a secret key agreement protocol that, for every pair of n-bit strings x and y, allows Alice and Bob to compute with high probability a shared secret key of length $I(x:y)$ (up to an $O(\log n)$ additive term).*

2. *No protocol can produce a longer shared secret key (up to an $O(\log n)$ additive term) with a big probability.*

The proofs of this theorem can be found in [e1], see Appendix. In what follows we only sketch the main ideas and techniques used in the proof.

Comments on the positive result. Part (1) of Theorem 34 claims that there is a communication protocol that permits to agree on a shared secret key of size $I(x:y)$. On the high level this protocol looks similar to the protocol from the simple geometric example discussed above:

Step 1. Alice sends to Bob a "fingerprint" p_1 of x of size $\approx \mathrm{C}(x \mid y)$;

Step 2. Bob uses y and the received fingerprint p_1 to find x;

Step 3. then both Alice and Bob independently compute another fingerprint $p_2 = p_2(x)$ of size $I(x:y)$, which is used as a common secret key.

The construction of the fingerprints guarantees that the adversary obtains virtually no information about the final value p_2. Observe that the value of p_1 becomes public, and this divulges $\mathrm{C}(x \mid y)$ bits of information about x. So it is not surprising that

$$\mathrm{C}(x) - \mathrm{C}(x \mid y) =^+ I(x:y)$$

bits of information in x remain confidential. However, to make this plan work we have to choose an appropriate construction of "fingerprints." Technically, we use the fingerprints from a suitable version of Muchnik's theorem on conditional descriptions (see Section 1.6).

Remark 18. Our communication protocols are not time-efficient, and this cannot be improved in general. The same time, for some particular types of correlation (for instance, for a pair of inputs with a bounded Hamming distance) the protocols can be improved. We can construct a polynomial-time computable protocol by using an efficient reconciliation technique from [73].

Some comments on the negative result. Part (2) of Theorem 34 claims that there is no way to obtain a longer common secret key. This statement (similar to many other "no-go" results in information theory) is based on a relevant information inequality.

To explain the ideas behind the proof, we simplify the setting and assume that Alice and Bob cannot use randomness (i.e., the communication protocol is deterministic). Let us simplify the problem even further and assume that Alice and Bob want to agree on a common secret key without any communication. Though in the general case this is impossible, Alice and Bob can do it for some very particular x and y. (For example, of the first halves of strings x and y are the same, then Alice and Bob can take this prefix shared by their inputs as a secret key.) In this case the problem of a secret key agreement reduces to the problem of extracting the common information from x and y: Alice and Bob need a key w such that w can be easily computed given x and given y.

We have seen in Section 1.8 that for any x, y the common information cannot be greater than the mutual information $I(x:y)$, i.e.,

$$\text{if } \mathrm{C}(w \mid x) \approx 0 \text{ and } \mathrm{C}(w \mid y) \approx 0, \text{ then } \mathrm{C}(w) \lesssim I(x:y). \tag{1.53}$$

Now we extend the setting and allow the communication between Alice and Bob. Denote t the *transcript* of the protocol (logging of the data sent by Alice and Bob to each other via the channel). Then the final result (the common secret key w) must be easily computed from x, t (this is done by Alice) and from y, t (this is done by Bob). The secrecy means that the obtained key has (almost) no mutual information with the public transcript t. The relativized version of (1.53) provides an upper bound for the size of this key:

$$\text{if } \mathrm{C}(w \mid x,t) \approx 0, C(w \mid y,t) \approx 0, \text{ and } I(w:t) \approx 0, \text{ then } \mathrm{C}(w) \lesssim I(x:y \mid t). \tag{1.54}$$

Thus, we end up with the bound $\mathrm{C}(w) \lesssim I(x:y \mid t)$, though we need $\mathrm{C}(w) \lesssim I(x:y)$. It remains to compare the values of $I(x:y \mid t)$ and $I(x:y)$.

We know that in general, the value of $I(x:y \mid t)$ can be much greater than $I(x:y)$ (see p. 35). However, in this particular case (when t is the transcript of a communication protocol), we can prove that $I(x:y \mid t) \le^+ I(x:y)$.

Lemma 6 ([e1, c1])**.** *If t is the transcript of a deterministic communication protocol for two parties that are given x and y as inputs, then $I(x:y:t) \ge^+ 0$, see Fig. 1.27.*

This lemma in some sense translates in the framework of Kolmogorov complexity the constraint information inequality from Proposition 4. It is instructive to compare Lemma 6 with negative results in Theorem 31 on p. 81 (so it is crucial that the constraint imposed on x, y, t in this lemma is not slack, not "up to a logarithmic term").

Somewhat implicitly Lemma 6 provides a statement on biclique covering of a bipartite graph. The vertices of this graph are n-bit strings x and y, and the bicliques are transcripts of the protocol (a biclique consists of all pairs (x,y) compatible with one specific instance of a transcript), see Section 1.13 for a discussion of the connection between communication complexity, biclique covering, and constraint information inequalities.

Lemma 6 together with the usual argument "common information is not greater than the mutual information" results in the required bound: Alice and Bob cannot agree on a common secret key with complexity greater than $I(x:y)$.

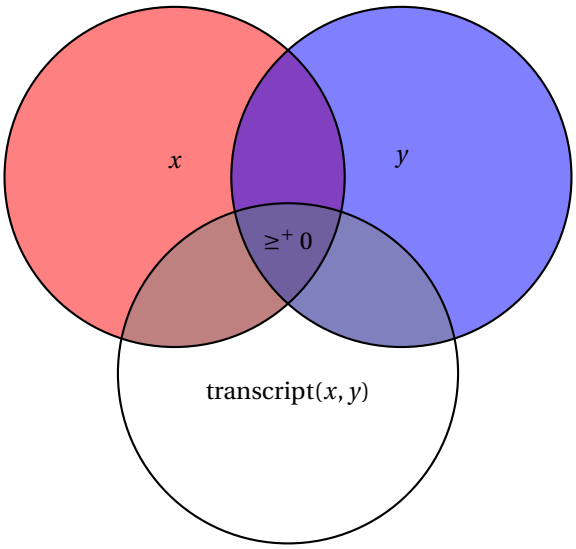

Figure 1.27: The diagram for x, y, and the transcript of a communication protocol with inputs x and y. Lemma 6 claims that $I(x:y) \geq^+ I(x:y \mid \text{transcript})$.

For the sake of simplicity, in this sketch we discarded the private randomness available to Alice and Bob. Of course, the general proof cannot ignore this issue. However, taking into account these additional random bits causes no serious technical difficulties.

Remark 19. For communication protocols with $O(1)$ rounds a version of Lemma 6 can be proven with the standard Shannon-type inequalities. The new technique is vital when the number of rounds grows with n.

Some comments on communication complexity of the secret key agreement. In the protocol suggested in Theorem 34 Alice and Bob agree on a common secret key of size $I(x:y) - O(\log n)$ by exchanging $C(x \mid y) + O(\log n)$ bits. With a symmetric version of this protocol, we obtain the communication complexity

$$\min(C(x \mid y), C(y \mid x)) + O(\log n) \tag{1.55}$$

bits. This protocol is randomized. We usually assume that Alice and Bob use their private sources of random bits, but the protocol can be adapted to the setting with public random bits (accessible to Alice and Bob, and to the adversary as well). For the communication model with a public source of randomness, we show that the communication complexity of the proposed protocol is optimal. More precisely, for any protocol with public randomness (that produces with high probability a common secret key of size $\approx I(x:y)$) there are pairs of inputs x and y on which Alice and Bob exchange at least $\min(C(x \mid y), C(y \mid x))$ bits (up to an additive logarithmic term). We prove even a stronger statement: if the communication complexity of a

protocol is below the threshold (1.55), then the size of the resulting common secret key is close to zero.

The proof of this result is based on the thorough analysis of the extractability of common information (see Section 1.8). More specifically, we use Theorem 22 on the common information for stochastic pairs.

The question on the optimal communication complexity for the model with private randomness remains open.

Secret key agreement for many parties Secret key agreement can be extended to the case of $\ell > 2$ parties. If ℓ parties are given n-bit strings $x_1, \ldots, x_\ell$ respectively and they known the complexity profile of this tuple of strings, then they can obtain a common secret key of length

$$\mathrm{C}(x_1, \ldots, x_\ell) - \mathrm{CO}(x_1, \ldots, x_\ell), \tag{1.56}$$

where $\mathrm{CO}(x_1, \ldots, x_\ell)$ denotes the minimum communication for the *omniscience* task in the broadcast model. (An *omniscience protocol* make sure that all parties get to known the entire tuple $x_1, \ldots, x_\ell$.) The proposed protocol is randomized, it returns a valid result with a probability close to 1. We show that the bound (1.56) is essentially tight (up to $O(\log n)$), [e1, c1].

Historical comments. The idea of *randomness* plays the central role in cryptography: an encrypted message must look for an eavesdropper as a random sequence of digits, a secret password must be chosen at random to make it hard to break, and so on. Let us consider a more specific example. In the one-time pad scheme, we assume that two parties share a "random" key, and this key is a "secret" for the attacker. It is assumed usually that both parties have access to a common source of randomness, e.g., to the results of tossing a fair coin (invisible to the adversary). The intuitive ideas of randomness and secrecy are usually formalized in cryptography in the framework of probability theory and Shannon's entropy. If we toss a fair coin n times, we obtain a random variable with maximal possible entropy n, and therefore, from the Shannon's entropy perspective, the quality of the resulting random key is perfect. However, if by chance we obtain a sequence that consists of n zeros (this outcome is possible, as well as any other), then this specific one-time pad looks useless in any practical application. Disappointingly, Shannon's information theory provides no language to complain about an apparently non-random *individual* key. This is the point where Kolmogorov complexity comes into play, since Kolmogorov complexity measure randomness in individual objects, not in a distribution in general. The idea to use Kolmogorov complexity to measure the secrecy of an individual instance of a one-time pad or a secret sharing schemes was suggested by Antunes *et al.* in [70]. While discussing the idea of the secret key agreement, we keep in mind a similar motivation: we understand a good "secret key" as an individual string that is incompressible in the sense of Kolmogorov complexity.

Let us focus now on the notion of the *secret key agreement*. The secret key agreement was extensively studied in classic (Shannon's) information theory. In this framework Alice gets as an input values of random variables $(X_1, \ldots, X_n)$, Bob gets

values of random variables $(Y_1, \ldots, Y_n)$, where the random pairs (X_i, Y_i) are i.i.d., or at least have the properties of stationarity and ergodicity. Then Alice and Bob communicate via a public channel. The aim of Alice and Bob is to agree on a common value of a random variable ("secret key") that would have large Shannon's entropy even conditional on the transcript of the protocol. Our work has been inspired by the results proven in this setting in [30], [31], [59].

Our results can be understood as algorithmic (Kolmogorov-type) analogs of the Shannon-type theorems proven in [30], [31], [59]. Technically speaking, the upper and lower bounds on the size of the common key that we have proven in the framework of Kolmogorov complexity, formally imply similar bounds previously known in Shannon's framework for stationary ergodic sources. Indeed, it is well known that a sequence of i.i.d. random variables (or, more generally, an outcome of a stationary ergodic source) gives with a high probability a string of letters whose Kolmogorov complexity is close to Shannon's entropy of the random source, [58], [76]. Hence, if Alice gets a value of $(X_1, \ldots, X_n)$, Bob gets a value of $(Y_1, \ldots, Y_n)$, and the sequence of pairs (X_i, Y_i) is a stationary ergodic source, then with high probability the mutual information (in the sense of Kolmogorov complexity) between Alice's and Bob's individual inputs is close to the mutual information between these two random sequences (in the sense of Shannon's entropy). The same time, our results are in a sense more general than their homologs in Shannon's theory: we do not need to assume the properties of memoryless or ergodicity of sources.

The proof of our positive results (the constructions of communication protocols) at the high level look similar to those from their Shannon's counterparts in [30], [31], [59]. In fact, we use similar intuitive ideas (that can be explained in terms of manipulations with fingerprints of appropriate lengths). However, the technical implementation of these ideas is pretty different in Kolmogorov's and Shannon's frameworks. When working with Kolmogorov complexity, we need quite explicit constructions (we use extractor graphs and universal hashing), while similar results for Shannon's entropy are typically obtained by a straightforward choice of random encodings.

In the proof of the negative results (upper bounds for the size of the common secret key), not only the technical implementation but also the intuitive ideas behind the proofs somewhat differ in Kolmogorov's and Shannon's frameworks. The reason for this discrepancy is that in Kolmogorov's framework, we cannot say that the mutual information between two objects is *exactly* zero. We can only say that it is close to zero, within some minor terms. Even though each minor term is negligibly small, the errors can accumulate during the rounds of a protocol, and become significant at the end of the protocol. To overcome this obstacle we proposed a new technique based on constraint inequalities for Kolmogorov complexity. We suppose that this technique can be eventually reemployed in Shannon's framework to achieve stronger no-go results in classical information theory.

To conclude, we summarize the techniques used in this section:

- In the proof of positive results (constructions of protocols) we used a variant of Muchnik's theorem on conditional descriptions (discussed in detail in Section 1.6).

- In the proof of negative results (the upper bound for the size of a common se-

cret key) we used conventional information inequalities and also some new constraint information inequalities and their combinatorial interpretation (see Section 1.4, Section 1.11 and Section 1.13). The proofs of the constraint information inequalities involve the technique of "clones," see Section 1.10.

- In the lower bound for the communication complexity of protocols, we used the results on extracting the common information (see Section 1.8).

The detailed proofs of all results discussed in this section can be found in [e1], see Appendix.

Chapter 2

Self-simulating tilings

2.1 Shifts and tilings: Introduction

In this chapter of the manuscript, we study the multi-dimensional *shifts*, i.e., the translation invariant and topologically closed sets of configurations of a Cantor space. More precisely, we deal with the translation invariant and topologically closed sets of configurations over a finite alphabet on $\mathbb{Z}^d$. Every shift can be specified by a set of *forbidden patterns* $\mathscr{F}$, so that a configuration belongs to the shift, if and only if, it does not involve any forbidden finite pattern from $\mathscr{F}$.

A trivial example: Let us take the two letters alphabet that consists, say, of a *white square* and a *black square*, $\Sigma = \{$□,■$\}$, and consider configurations over Σ on $\mathbb{Z}^2$. Let the forbidden patterns be all adjacent pairs of squares with the same colors: ■■, □□, ▪, and ▫. Then the configurations avoiding these patterns are the checkerboard colorings of $\mathbb{Z}^2$, as shown in Fig. 2.1. Technically, there are exactly two valid configurations, which differ from each other by a translation. These two infinite configurations form a shift.

The notion of a shift is an abstraction that helps us to study how simple local rules can imply nontrivial global phenomena. Shifts are used in various domains: in mathematical logic (in problems of decidability of formal languages with very simple syntax, see, e.g., [39]), in computational complexity (as a computational model with nice structural properties, [20]), in mathematical physics (as a simplistic model of quasicrystalline structures, see, e.g., [28]), and so on.

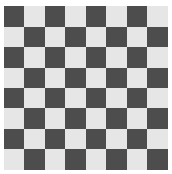

Figure 2.1: A checkerboard configuration.

Two classes of shift play a prominent role in symbolic dynamics, in language theory, and in the theory of computability: *shifts of finite type* (obtained by forbidding a finite number of finite patterns) and *effective shifts*, otherwise known as *effectively closed shifts* (obtained by forbidding a computable set of finite patterns). The class of shifts of finite type is known to be very rich — even a finite set of simple local rules can imply rather sophisticated global properties. It is known that some (non-empty) multi-dimensional shifts of finite type admit only aperiodic (see [4]) or even only non-computable ([15], [16]) configurations. The value of the combinatorial entropy of a shift of finite type can be any right computable real number $h \geq 0$, see [82]. The projective subdynamics of a shift of finite type of dimension $d > 1$ can be any effective shift of a dimension below d (with possibly infinitely many forbidden patterns), see [80], [j7], [88].

Most proofs of the results mentioned above are based on embedding a computation in the structure of a shifts of finite type. In this part of the manuscript we discuss one specific technique of embedding a Turing machine in a shift of finite type. We refer to this technique as *self-simulating* or *fixed-point tilings*. This method combines the idea of geometric self-similarity with the idea of a self-referential program (in a very general sense). The idea of using a program that manipulates its own text is by no means new. The first applications of this method arguably date back to the classic recursion theorems by Kleene and Gödel's incompleteness theorems. Nowadays the technique of self-referential programs is very common in computer science. Writing exotic *quines* (self-replicating programs that can print a copy of its own source code) in different programming languages has become a particularly popular ludic activity for young programmers. And, of course, this idea appears in various contexts in numerous serious applications.

We use the technique of self-referential programming in a form similar to the construction of self-reproducing automata by J. von Neumann, [38], and to the self-correcting cellular automata by P. Gács, [24]. It turns out that the idea of a self-referential program matches perfectly the idea of a self-similar geometric configuration that reproduces similar patterns in different scales. Using this technique we give new proofs of several classic theorems of symbolic dynamics and then prove several new results. In what follows we briefly mention the results that we obtain.

Simple local rules can imply complex global behavior.

- For every $d \geq 2$ there exists a non-empty SFT of dimension d where all configurations are aperiodic. This is a classic result originally proven by Berger in the seminal paper [4]. The very fact of the existence of aperiodic SFTs is nontrivial and has important implications. What is even more important, aperiodic SFTs are used as a basis to construct shifts with various other nontrivial properties.

  Berger used an SFT defined in terms of *Want tiles* (i.e., each letter of the alphabet is a square with colored sides, and the matching rules of this SFT require that every two neighboring squares share the same color on the common side, see the formal definition below). Berger's proof involved the idea of self-similarity: each admissible configuration must be in some non-literal sense self-similar, and therefore aperiodic.

The original construction proposed by Berger was very technical. Several simpler proofs of this result have been suggested by other authors in subsequent works. Some of these proofs were based on simplified versions of Berger's construction ([9]; see also [39], [62], [63]). An elegant and simple proof based explicitly on ideas of self-similarity was proposed in [77]. A series of examples based on the idea hierarchical structures were constructed in [96]. Some other approaches were based on substitutive polygonal tilings (such as the Penrose and Ammann tilings; see [25]). A very different construction suggested in [37] was based on multiplication in a sort of positional number system; this approach gave a very small aperiodic set of only 14 Wang tiles ([36] presented an improved version with 13 tiles and [94] with only 11 tiles, which is the minimal possible set of Wang tiles with the property of aperiodicity).

The known constructions of aperiodic SFT combine in different proportions geometric and algorithmic ideas. We propose one more construction, which is in some sense extremal: it is almost purely algorithmic, with minimum of geometric tricks. Our technique does not help minimize the size of the alphabet or reduce the number of local constraints in an SFT. However, this construction gives much more than just another example of an aperiodic SFT. The advantage of this method is its flexibility. As we explain below, it allows various modifications and extensions, so we can adjust it to many applications.

- The result on aperiodic SFT can be made stronger: we show that for every $d \geq 2$ there exists a non-empty SFT of dimension d such that for every configuration, every non zero translation changes a constant fraction of points. (Note that the conventional property of aperiodicity implies only that every translation changes at least some points of every configuration, and the fraction of the changed points can tend to 0 as the length of the translation vector goes to infinity).

- We show that for every $d \geq 2$ there exists a non-empty SFT where all configurations are uncomputable. This is another classic result originally proven in [15], [16]. We give a new proof of this theorem as an illustration of the flexibility of our technique.

Multi-dimensional SFTs can have highly complex subdynamics.

- Every effective shift on $\mathbb{Z}^d$ can be represented as a projection of subdynamics in an SFT on $\mathbb{Z}^{d+1}$. Originally, a slightly weaker version of this result (for SFT on $\mathbb{Z}^{d+2}$) was proven by Hochman, [80]. The present version of the theorem was independently proven in [j7] (with the method of self-simulating tilings) and with a more conventional technique of Robinson's tilings in [88].

- For every $d \geq 2$ there exists a non-empty SFT of dimension d where for all configurations each $n \times n \times \cdots \times n$ pattern has Kolmogorov complexity $\Omega(n^{d-1})$. This statement was originally proven by Durand, Levin, and Shen, [75], with a technique based on Robinson's tilings. We prove the same result with self-simulating tilings.

Restrictive classes of SFT: quasi-periodic and minimal shifts can have complex algorithmic and dynamical properties.

- For every $d > 1$ there exists a non-empty SFT on $\mathbb{Z}^d$, where each configuration is at once aperiodic and quasiperiodic (and even minimal). This statement was originally proven in [78] for an SFT constructed in [77], via a very thorough analysis of a Robinson-type construction. We suggest a pretty different proof, based again on self-simulating tilings.
- For every $d > 1$ there exists a non-empty SFT on $\mathbb{Z}^d$ where each configuration is at once non-computable and quasiperiodic. We also prove a stronger result and characterize the classes of Turing degrees that can correspond to the configurations of a quasi-periodic SFT. To the best of our knowledge, the only known proof of these results uses our technique of self-simulating tilings.

Constructions of robust (fault-tolerant) SFTs. We extend several of the results mentioned above (aperiodic and strongly aperiodic SFT, SFT with patterns with high Kolmogorov complexity) to a more general type of objects — to so called *faulty tilings*. The faulty tilings are the configurations where the local rules are not absolute anymore: the constraints can be violated on a sparse enough set of "errors" (we give a formal definition below). We say that a set of local rules is fault-tolerant, if some global properties of configurations (strong aperiodicity, high complexity) remain valid despite sparse departures from these local rules. It turns our that the technique of self-simulating tilings is a perfect method to design fault-tolerant tilings.

Historical remarks. The technique of self-simulating tilings was largely inspired by the construction of reliable cellular automata with self-organization suggested by P. Gács. The self-simulating tilings first appeared in the series of work [c9, c7, j7] (a joint work with Bruno Durand and Alexander Shen), where the principal motivation was to address the "faulty tilings." In [c4, c2] (a joint work with Bruno Durand) this technique was used to study minimal and quasiperiodic SFT; in [e2] the same method was applied to study transitive SFT. Similar constructions of self-simulating tilings were used in [101] and [95], [98].

2.2 Notation and basic definitions

In this section we recall several standard definitions from symbolic dynamics and introduce some notation.

Shifts. Let Σ be a finite set (an alphabet). Fix an integer $d > 0$. A Σ-*configuration* (or just a *configuration* if Σ is clear from the context) on $\mathbb{Z}^d$ is a mapping $\mathbf{f} : \mathbb{Z}^d \to \Sigma$, i.e., a coloring of $\mathbb{Z}^d$ by "colors" from Σ. A $\mathbb{Z}^d$-*shift* (or just a *shift*) is a set of configurations that is (i) translation invariant (with respect to the translations along each coordinate axis), and (ii) closed in Cantor's topology. The entire space $\Sigma^{\mathbb{Z}^d}$ is itself a shift, referred to as *the full shift*.

A *pattern* is a mapping from a finite subset of $\mathbb{Z}^d$ to Σ (a coloring of a finite set of $\mathbb{Z}^d$); this set is called the support of the pattern. We say that a pattern P *appears* in a configuration $\mathbf{f}(\bar{x})$ if for some $\bar{c} \in \mathbb{Z}^d$ the pattern P coincides with the restriction of the shifted configuration $\mathbf{f}_{\bar{c}}(\bar{x}) := \mathbf{f}(\bar{x} + \bar{c})$ to the support of this pattern.

Every shift is determined by the corresponding set of forbidden finite patterns $\mathscr{F}$ (a configuration belongs to the shift if and only if no patterns from $\mathscr{F}$ appear in this configuration).

A shift is called *effective* (or *effectively closed*) if it can be defined by a computably enumerable set of forbidden patterns. A shift is called a *shift of finite type* (SFT) if it can be defined by a finite set of forbidden patterns.

Wang tilings. A special class of two-dimensional SFTs is defined in terms of *Wang tiles*. In this case, we interpret the alphabet Σ as a set of *tiles*, i.e., a set of unit squares with colored sides, assuming that all colors belong to some finite set C (we assign one color to each side of a tile, so technically Σ is a subset of C^4). A (valid) *tiling* is a set of all configurations $\mathbf{f} : \mathbb{Z}^2 \to \Sigma$ where every two neighboring tiles match, i.e., they share the same color on adjacent sides. Wang tiles are powerful enough to simulate any SFT in a very strong sense: for each SFT $\mathscr{S}$ there exists a set of Wang tiles τ such that the set of all τ-tilings is isomorphic to $\mathscr{S}$. This definition has a straightforward generalization for all dimensions $d > 2$.

In this paper we mainly use the formalism of tilings since Wang tiles are better adapted for explaining our techniques of self-simulation.

A shift as a dynamical system. Every shift $\mathscr{S} \subset \Sigma^{\mathbb{Z}^d}$ can be interpreted as a dynamical system. Indeed, there are d translations along the coordinate axes, and each of these translations maps $\mathscr{S}$ to itself. Therefore, the group $\mathbb{Z}^d$ naturally acts on $\mathscr{S}$.

Let $\mathscr{S}$ be a shift on $\mathbb{Z}^d$ and L be k-dimensional sub-lattice in $\mathbb{Z}^d$ (i.e., L must be an additive subgroup of $\mathbb{Z}^d$ that is isomorphic to $\mathbb{Z}^k$). Then the *L-projective subdynamics* $\mathscr{S}_L$ of $\mathscr{S}$ is the set of configurations of $\mathscr{S}$ restricted on L. The L-projective subdynamics of a $\mathbb{Z}^d$-shift can be understood as a $\mathbb{Z}^k$-shift (note that L naturally acts on $\mathscr{S}_L$). In particular, for every $d' < d$ we have a $\mathbb{Z}^{d'}$-projective subdynamics on the shift $\mathscr{S}$, generated by the lattice on the first d' coordinate axis.

A configuration $\mathbf{x}$ is called *recurrent* if every pattern that appears in $\mathbf{x}$ at least once, must then appear in this configuration infinitely often.

Subdynamics of a shift. We will prove several results concerning the subdynamics of an SFT. More specifically, we will prove that SFTs can "simulate" in some sense any effective shifts in the lower dimensions. By simulation we mean the projective subdynamics of a shift on $\mathbb{Z}^{d+1}$ that corresponds to the restriction of a shift on the first d coordinate axis.

Definition 13. *We say that a shift $\mathscr{A}$ on $\mathbb{Z}^d$ is* simulated *by a shift $\mathscr{B}$ on $\mathbb{Z}^{d+1}$, if there exists a projection $\pi : \Sigma_B \to \Sigma_A$, such that for every configuration*

$$\boldsymbol{f} : \mathbb{Z}^{d+1} \to \Sigma_B$$

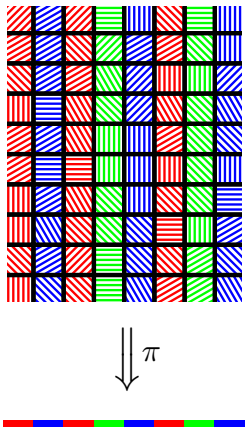

Figure 2.2: In this example, each cell of a two-dimensional configuration has two characteristics: the color and the direction of hatching. The color is maintained unchanged along each vertical line. The projection π maps each column to its color.

from $\mathscr{B}$ and for all $i_1, \ldots, i_d, j, j'$ we have $\pi(\mathbf{f}(i_1, \ldots, i_d, j)) = \pi(\mathbf{f}(i_1, \ldots, i_d, j'))$ (i.e., projection π takes a constant value along each column $(i_1, \ldots, i_d, *)$, see Fig. 2.2), and the resulting d-dimensional configuration

$$\{\pi(\mathbf{f}(i_1, \ldots, i_d, *))\}$$

belongs to $\mathscr{A}$; moreover, each configuration of $\mathscr{A}$ can be represented in this way by some configuration of $\mathscr{B}$. Informally, we can say that each configuration from $\mathscr{B}$ encodes a configuration from $\mathscr{A}$, and each configuration from $\mathscr{A}$ is encoded by some configuration from $\mathscr{B}$.

Quasiperiodicity and minimality. A configuration $\mathbf{x} : \mathbb{Z}^d \to \Sigma$ is called *quasiperiodic* (or *uniformly recurrent*) if every pattern P that appears in $\mathbf{x}$ at least once must appear in every large enough d-dimensional hypercube Q in $\mathbf{x}$. Note that every periodic configuration is also quasiperiodic. A *quasiperiodic shift* is a shift that contains only quasiperiodic configurations. By compactness, if a shift is minimal then it is quasiperiodic (the converse is not true).

Given a configuration $\mathbf{x}$, a *function of a quasiperiodicity* for $\mathbf{x}$ is a mapping $\varphi : \mathbb{N} \to \mathbb{N} \cup \{\infty\}$ such that every finite pattern of size (diameter) n either never appears in $\mathbf{x}$ or appears in every hypercube of size $\varphi(n)$ in $\mathbf{x}$ (see [46]). We assume $\phi(n) = \infty$ if some pattern P of size n appears in $\mathbf{x}$, but there exist arbitrarily large areas in $\mathbf{x}$ that are free of P. By definition, for a quasiperiodic $\mathbf{x}$ we have $\phi(n) < \infty$ for all n. We say that a shift $\mathscr{S}$ has a function of quasiperiodicity φ if φ is a function of quasiperiodicity for every configuration in $\mathscr{S}$.

If a shift $\mathscr{S}$ is minimal, then all configurations in $\mathscr{S}$ have exactly the same finite patterns and thus the same functions of quasiperiodicity. Every minimal shift $\mathscr{S}$ is quasiperiodic, so its function of quasiperiodicity must therefore be finite (for every n). Moreover, for an effective minimal shift, the set of all finite patterns (that can appear in any configuration) is computable, see [80], [81]. From this fact it follows that every effective and minimal shift contains some computable configuration. Indeed, with an algorithm that checks whether a given pattern appears in every $\mathbf{x} \in \mathscr{S}$, we can incrementally (and algorithmically) increase a finite pattern, maintaining the property that this pattern appears in every configuration in $\mathscr{S}$.

2.3 General remarks and organization of this part of the manuscript

In most theorems in this chapter, we claim something about (quasiperiodic, minimal, transitive) SFTs. In the proofs we deal mostly with tilings, which are a very special type of SFT. Since the principal results of the paper are positive statements (we claim that SFTs with some specific properties do exist), the focus on tilings does not restrict the generality. On the other hand, the formalism of Wang tiles matches very well the constructions of self-similar and self-simulating shifts of finite type, which represent the main technique used in this chapter To simplify the notation and make the argument more visual, in what follows we focus on the case $d = 2$. The proofs extend to any $d > 1$ in a straightforward way, *mutatis mutandis*.

The central idea of our arguments is the notion of self-simulation. The technique of hierarchical self-simulating tilings is quite generic and flexible. The drawback of this approach is that it is hard to isolate the core technique from features specific to some particular application. In every specific result we cannot just cite the statement of a previously known theorem about self-simulating tilings: rather, we need to reemploy the constructions from the proofs of a previously known theorem (embedding some new gadgets in the previously known scheme). This make the proofs long and somewhat cumbersome. So, in order to help the reader, we made the exposition of the main results "hierarchical." We start with a general perspective of our technique (illustrating it by proofs of several classic results). and then, in each succeeding section, we show how to adjust and extend this general construction to prove this or that particular result.

2.4 The generic framework of self-simulating SFT

In this section, we recall the principal elements of the technique of *self-simulating tile sets*. We start with a very basic version of a construction of self-simulating tile sets from [j7]. This part of the construction will be enough, in particular, to obtain a proof of the classic Berger Theorem:

Theorem 35 (Berger, [4])**.** *For every $d > 1$ there exists a non-empty SFT on $\mathbb{Z}^d$, where each configuration is aperiodic.*

Later, we will extend this construction and adapt it to prove much stronger statements.

2.4.1 The simulation relation for tile sets

Let τ be a tile set and $N > 1$ be an integer. We call a *macro-tile* an $N \times N$ square tiled by matching tiles from τ. Every side of a τ-macro-tile contains a sequence of N colors (of tiles from τ); we refer to this sequence as a *macro-color*. Furthermore, let T be a set of τ-macro-tiles (of size $N \times N$). We say that τ *implements* T with a *zoom factor* N if

- some τ-tilings exist, and
- for every τ-tiling there exists a unique lattice of vertical and horizontal lines that cuts this tiling into $N \times N$ macro-tiles from ρ.

A tile set τ *simulates* another tile set ρ, if τ implements a set of macro-tiles T (with a zoom factor $N > 1$) that is isomorphic to ρ, i.e., there exists a one-to-one correspondence between ρ and T, such that the matching pairs of ρ-tiles correspond exactly to the matching pairs of T-macro-tiles. A tile set τ is called *self-similar* if it simulates itself.

If a tile set τ is self-similar, then all τ-tilings have a hierarchical structure. Indeed, each τ-tiling can be uniquely split into $N \times N$ macro-tiles from a set T, and these macro-tiles are isomorphic to the initial tile set τ. Further, the grid of macro-tiles can be uniquely grouped into blocks of size $N^2 \times N^2$, where each block is a macro-tile of rank 2 (again, the set of all macro-tiles of rank 2 is isomorphic to the initial tile set τ), etc. It is not hard to deduce that a self-similar tile set τ has only aperiodic tilings (for more details, see [j7]). Below, we discuss the generic construction of self-similar tile sets.

2.4.2 Simulating a tile set defined by a Turing machine

Let us have a tile set ρ. In what follows, we present a general construction that allows to simulate ρ by some other tile set τ, with a large enough zoom factor N. The number of tiles in the simulating tile set τ will be $O(N^2)$, and the constant in the $O(\cdot)$-notation does not depend on the simulated ρ.

We assume that each color is a string of k bits (i.e., the set of colors $C \subset \{0,1\}^k$) and the set of tiles $\rho \subset C^4$ is presented by a predicate $P(c_1, c_2, c_3, c_4)$ (the predicate is true if and only if the quadruple (c_1, c_2, c_3, c_4) corresponds to a tile from ρ). Suppose we have a Turing machine $\mathcal{M}$ that computes P. (It might look wasteful to construct a Turing machine that computes a predicate with a finite domain, but we shall see that this kind of abstraction is useful.) Now we construct in parallel a tile set τ and a set τ-macro-tiles that simulate the given ρ.

When constructing a tile set τ, we keep in mind the desired structure of τ-macro-tiles (that should simulate the given tile set ρ). We require that each tile in τ "knows" its coordinates modulo N in the tiling. This information is included in the tile's colors. More precisely, for a tile that is supposed to have coordinates (i, j) modulo N,

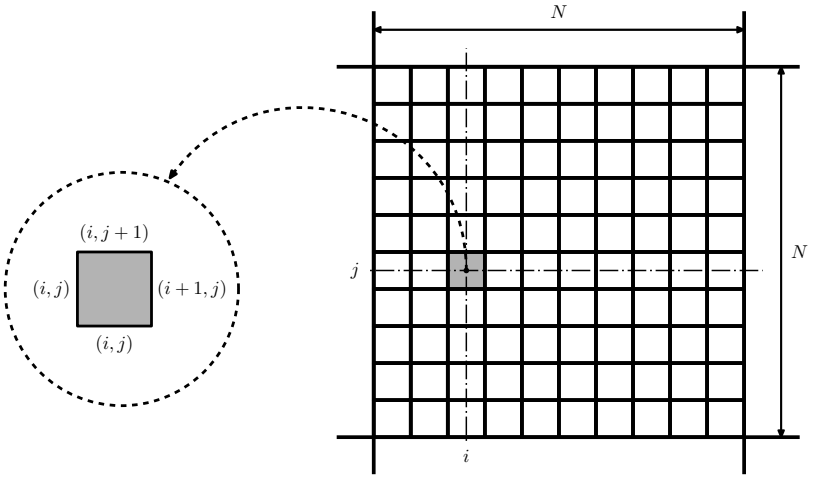

Figure 2.3: The basic structure of a macro-tile: A block of size $N \times N$ consists of tiles, the coordinates of which in this block (a pair of integers between 0 and $N-1$) are written as "colors" of the left and the bottom side of each tile. On the right and bottom sides of every tile, one of the coordinates is incremented (modulo N).

the colors on the left and on the bottom sides should involve (i, j), the color on the right side should involve $(i+1 \mod N, j)$, and the color on the top side involve $(i, j+1 \mod N)$, see Fig. 2.3.

This means that every τ-tiling can be uniquely split into blocks (macro-tiles) of size $N \times N$, where the coordinates of the cells range from $(0,0)$ in the bottom-left corner to $(N-1, N-1)$ in top-right corner, as shown in Fig. 2.3. Intuitively, each tile "knows" its position in the corresponding macro-tile. We will require that in addition to the coordinates, each tile in τ has some supplementary information encoded in the colors on its sides. On the border of a macro-tile (where one of the coordinates is zero) we assign to the colors of tiles one additional bit of information. Thus, for each macro-tile of size $N \times N$ the corresponding macro-colors can be represented as strings of N zeros and ones. We assume that $N \gg k$. We allocate k positions in the middle of a macro-tile's sides and make them represent colors from C; the other $(N-k)$ bits on the sides of a macro-tile are set to zero.

We now introduce additional restrictions on the tiles in τ that will guarantee that the macro-colors on the macro-tiles satisfy the "simulated" relation P. To this end, we ensure that bits from the macro-tile side are transferred to the central part of the tile, and the central part of a macro-tile is used to simulate a computation of the predicate P. We fix which cells in a macro-tile are "communication wires" and then require that these tiles carry the same (transferred) bit on two sides, see Fig. 2.4.

The central part of a macro-tile (of size, say, $m \times m$, where $m \ll N$) should represent a space-time diagram of the machine $\mathscr{M}$ (the tape is horizontal, and time goes up), see Fig. 2.5. This Turing machine processes the quadruple of inputs, which are

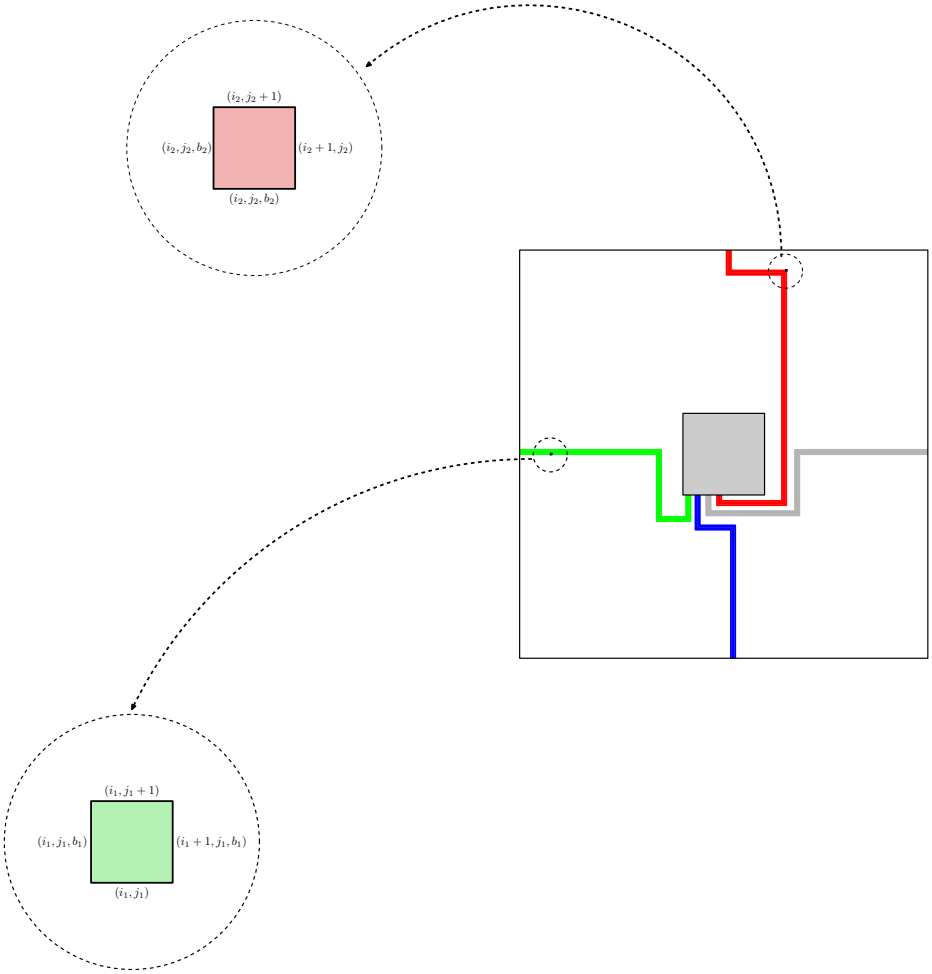

Figure 2.4: The tile with coordinates (i_1, j_1) is a part of a "communication wire" that transfers the bit b_1 (a horizontal part of the wire, the value of the bit b_1 is conducted from the left to the right), and the tile with coordinates (i_2, j_2) is a part of a "communication wire" that transfers the bit b_2 (a corner of the wire, with the value of the bit b_2 conducted from the left side to the bottom side of the tile). The coordinates and the values of the transferred bits are embedded in the colors of the tiles' sides.

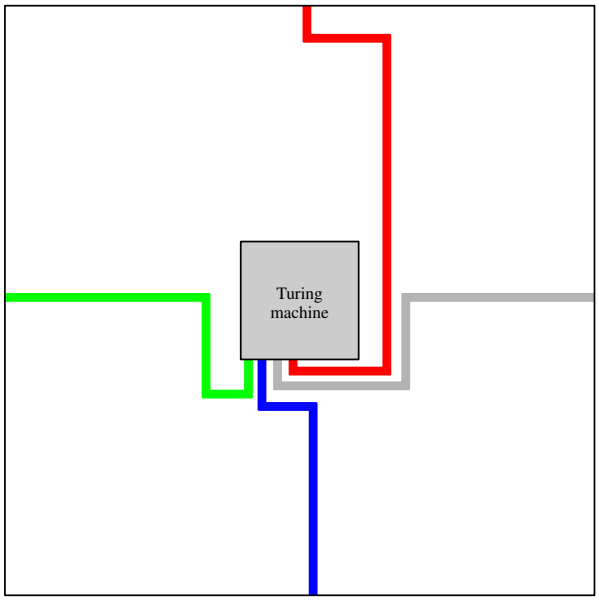

Figure 2.5: A macro-tile with a space-time diagram of a Turing machine in the middle part.

the k-bit strings representing the macro-colors of this macro-tile.

Let us explain in more detail how we represent the computation of a Turing machine in a tiling. Such a representation can be implemented in many different ways, but for the sequel we should fix one specific version. First of all, we assume that the machine has a single tape. We understand the *space-time diagram* of a Turing machine in a pretty standard way, as a table where each vertical column corresponds to one cell on the tape of the machine, and each horizontal row of the diagram represents an instance of a Turing machine configuration. For each row of the diagram, we

- specify for each cell (within a bounded part of the tape) the letter written in this position on the tape,
- specify with a special mark the position of the read head, and
- write the index of the internal state of the machine into the cell where the read head is currently located.

Each next row of the diagram represents the configuration of the machine at the next step of computation (once again, we assume that time goes up). Thus, the entire diagram is determined by its bottom line (with the input data of the machine).

The property of being a valid space-time diagram is defined locally, so we can easily represent such a diagram of a given Turing machine by local matching rules

for tiles. The details of this representation are not very important in the sequel; we may take, for example, the representation described in [39]. In what follows, we need only keep in mind some natural properties of the chosen representation (which hold trivially for the representations from [39]):

- a correct tiling of a frame $m \times m$ represents a space-time diagram of the same[1] size $m \times m$,
- a correct tiling of a frame $m \times m$ with some specific bottom line can be formed if and only if the computation (with the corresponding input data) terminates in an accepting state in at most m steps and during this computation the read head never leaves the available finite part of the tape,
- every $(k \times k)$-fragment[2] of a correct tiling can be reconstructed by its borderline (this is the only property where we need the Turing machine to be deterministic).

The communication of the computation zone with the "outside world" is restricted to the bottom line (the input data of the computation), which must cohere with the bits representing the four macro-colors of the macro-tile.

To make all of this construction work, the size of a macro-tile (the integer N) should be large enough: first, we need enough room to place the "communication wires" that transfer the bits of macro-colors to the "computation zone"; second, we need enough time and space in the computation zone of size $m \times m$ so that all accepting computations of $\mathcal{M}$ terminate in time m and on space m.

In this construction, the number of additional bits encoded in the colors of the tiles depends on the choice of machine $\mathcal{M}$. To avoid this dependency, we replace $\mathcal{M}$ with a fixed universal Turing machine $\mathcal{U}$ that runs a program simulating $\mathcal{M}$. Moreover, we prefer to separate the general program of a Turing machine (which involves a description of the predicate P corresponding to the simulated tile set ρ) from the zoom factor N.

Technically, we assume that the tape of the universal Turing machine has an additional read-only layer. Each cell of this layer carries a bit that never changes during the computation (so, in the computation zone, the columns carry unchanged bits). The construction of a tile set guarantees that these bits form two *read-only input fields*: (i) the program for $\mathcal{M}$ and (ii) the binary expansion of an integer N (which is interpreted as the zoom factor). Accordingly, the computation zone of a macro-tile represents a view of an accepting computation for that program given N as one of the inputs, see Fig. 2.6.

Thus, from now on, we assume that the simulated program is given five inputs: the binary codes of four macro-colors (that are "transferred" from the sides of this macro-tile) and the binary expansion of an integer N (interpreted as the zoom factor).

[1] In fact, it is enough to assume that a tiling of size $m \times m$ represents a space-time diagram of size $\Omega(m) \times \Omega(m)$.

[2] In what follows we use only a restricted version of this property. We need to be able to reconstruct every (2×2)-block of tiles given the 12 tiles around this block, see Fig. 2.14 below.

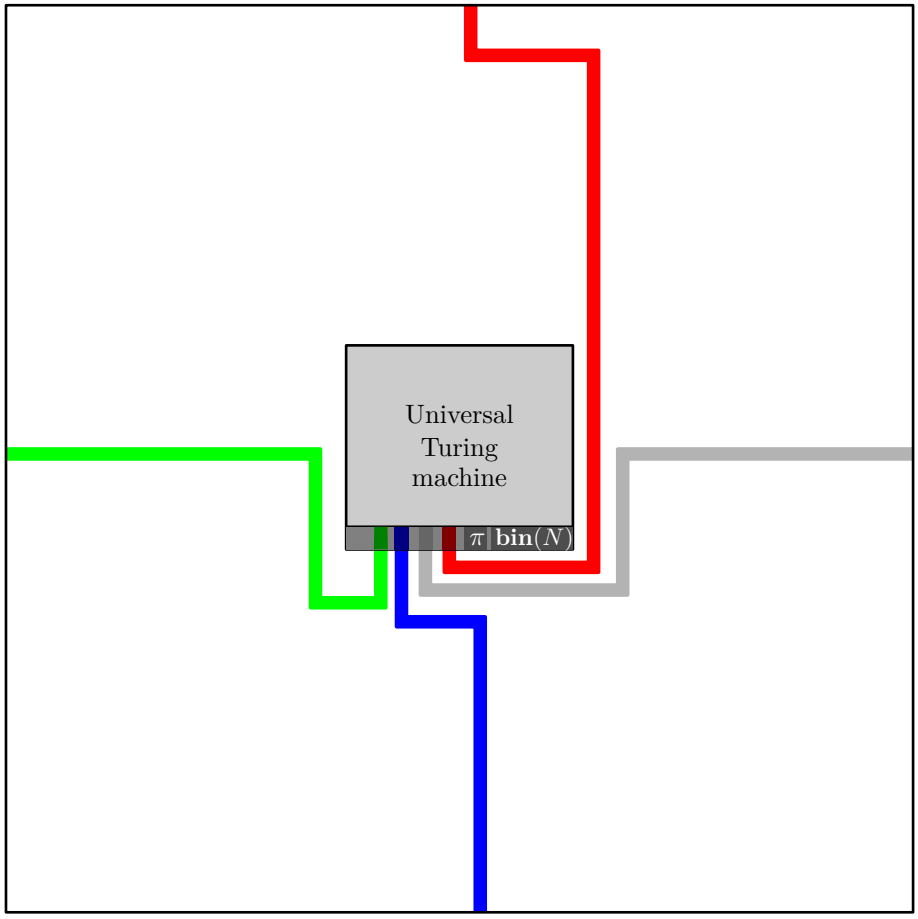

Figure 2.6: The computation zone represents a space-time diagram of the universal Turing machine. This machine simulates program π, which gets as an input the binary codes of four macro-colors, the binary expansion of the zoom factor N, and its own text.

Without loss of generality, we assume that the positions of the "wires" and the size of the "computation zone" in a macro-tile are chosen in some simple and natural way, and can be effectively computed given the size of the macro-tile N. Moreover, we may assume that the "geometry" of a macro-tile (the positions of the communication wires and of the computation zone) can be computed in polynomial time.

That is, given the binary expansions of integers N, i, j, we can compute in time poly($\log N$) the "role" played by a tile with coordinates (i, j) in a macro-tile of size $N \times N$ (there are several different "roles" for tiles: some of them are parts of communication wires, some others are parts of a computation zone, the others do not belong to any particular gadget; besides, for each tiles in a computation zone we can compute the *read-only input fields* assigned to the corresponding cell in the tape of the Turing machine).

In this way, we obtain an explicit construction of a tile set τ that has $O(N^2)$ tiles and simulates ρ. This construction works for all large enough N. The tile set τ depends on the program simulated in the computation zone and on the choice of zoom factor N. However, this dependency is very limited. The simulated program (and, implicitly, the predicate P) affects only the rules for the tiles used in the bottom line of the computation zone. The colors on the sides of all other tiles are generic and do not depend on the simulated tile set ρ.

2.4.3 Self-simulation with magical Kleene's recursion trick

We have explained how to implement a given tile set ρ by another tile set τ with a large enough zoom factor N. Now we want τ be isomorphic to ρ. This can be done using a trick similar to the proof of Kleene's recursion theorem. Roughly speaking, we employ the idea that a program can somehow *access its own text* and use its bits in the computation. On first viewing, this might look like a paradox (somehow, we need to know the code of the program before we write it), but technical implementation of this method is quite standard. It is especially transparent when we take the Turing machines as the model of computation.

Note that most steps in the construction of τ do not depend on the program for $\mathscr{M}$. Let us fix these rules as a part of ρ's definition and set $k = 2\log N + O(1)$, so that we can encode $O(N^2)$ colors by k bits. From this definition we obtain a program π that takes N as an input and checks that macro-tiles behave like τ-tiles in this respect. We are almost done with the program π. The only remaining part of the rules for τ is the hardwired program. We need to guarantee that the computation zone in each macro-tile carries the very same program π. But since the program (the list of instructions interpreted by the universal Turing machine) is written on the tape of the universal machine, this program can be instructed to access the bits of its own "text" and check that if a macro-tile belongs to the computation zone, this macro-tile carries the correct bit of the program.

It remains to choose the parameters N and m. We need them to be large enough so that the computation described above (which deals with inputs of size $O(\log N)$) can fit in the computation zone. The computations are rather simple (polynomial in input size, i.e., polynomial in $O(\log N)$), so they certainly fit in the space and time

bounded by $m = \text{poly}(\log N)$. Thus, we set $m(N) = \text{poly}(\log N)$ for some specific polynomial that is not too small (e.g., $m := (\text{poly}N)^3$ is enough) and choose N large enough so that $m(N) \ll N$, and the geometry of a macro-tile explained above can be realized. This completes the construction of a self-similar aperiodic tile set. Now, it is not hard to verify that the constructed tile sets (i) allow a tiling of the plane, and (ii) each tiling is self-similar.

The construction described above works well for all large enough zoom factors N. In other words, for all large enough N we get a self-similar tile set τ_N, and the tilings for all of these τ_N have very similar structure, with macro-tiles as shown in Fig. 2.6. Technically, program π (simulated by the universal Turing machine) now takes as its input a tuple of six strings of bits: the bit strings of length $k = k(N)$ representing the four macro-colors of a macro-tile, the binary expansion of the zoom factor N, and its own text. This program checks whether the given strings are coherent, i.e., whether the given quadruple of macro-colors in fact represents a quadruple of colors of one tile in our self-similar tile set τ_N (corresponding to the given value N of the zoom factor).

The presented construction of a self-simulating tile set provides a proof of Theorem 35, see a comment at the end of Section 2.4.1 and Fig. 2.7. In what follows, we extend and generalize this construction step-by-step, and then we apply it to prove much stronger statements.

Notation. We now introduce some useful terminology. In a hierarchical structure of macro-tiles, if a k-level macro-tile M is a "cell" in a $(k+1)$-level macro-tile M', we refer to M' as the *father* of M. We refer to the $(k+1)$-level macro-tiles neighboring M' as the *uncles* of M.

2.5 SFT with non-computable configurations

So far we have used the method of self-simulating tilings to enforce the property of aperiodicity. We now go further and construct an SFT where all configurations are with non-computability. To this end we introduce two technical tricks: self-simulating tilings with variable zoom factors and embedding a (non-computable) sequence of bits in a tiling.

2.5.1 Flexible zoom factors

For a large class of sufficiently "well-behaved" sequences of integers N_k we can construct a family of tile sets τ_k ($i = 0, 1, \ldots$) such that each τ_{k-1} simulates the next τ_k with the zoom factor N_k (and, therefore, τ_0 simulates each τ_k with the zoom factor $L_k = N_1 \cdot N_2 \cdots N_k$).

The idea is to reuse the basic construction from the previous section and vary the sizes of the macro-tiles (the zoom factors) on the different levels of the hierarchy. While in the basic construction the macro-tiles (built of $N \times N$ tiles), the macro-macro-tiles (built of $N \times N$ macro-tiles), the macro-macro-macro-tiles (built of $N \times N$ macro-macro-tiles), and so on, behave in exactly the same way, in the revised construction the behavior of a k-level macro-tile depends on k. We want to have

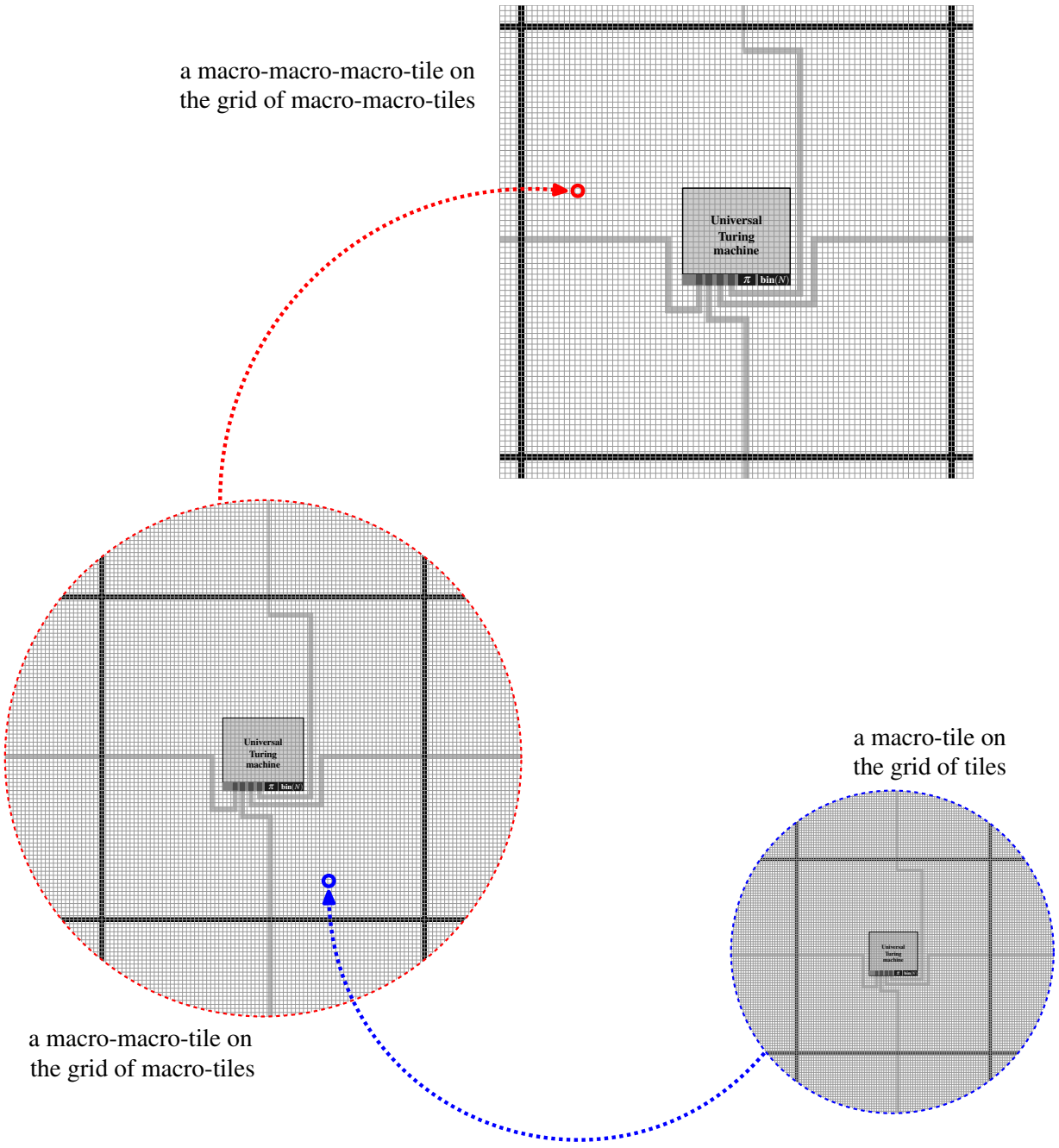

Figure 2.7: Hierarchical structure of macro-tiles. The k-level macro-tiles are blocks in the $(k+1)$-level macro-tiles, the $(k+1)$-level macro-tiles are blocks in $(k+2)$-level macro-tiles, etc. On all levels of the hierarchy, the structure of the macro-tiles is pretty much the same.

macro-tiles built of $N_1 \times N_1$ ground-level tiles, macro-macro-tiles built of $N_2 \times N_2$ macro-tiles, macro-macro-macro-tiles built of $N_3 \times N_3$ macro-macro-tiles, and so on. In this construction, the k-level macro-tiles will be isomorphic to the tiles of τ_k, and the idea of "self-simulation" should be understood less literally.

To implement this idea, we need only a minor revision of the construction from the previous section. Similar to our basic self-simulation construction, each tile of τ_k "knows" its coordinates modulo N_k in the tiling: the colors on the left and on the bottom sides should involve (i, j), the color on the right side should involve $(i+1 \mod N_k, j)$, and the color on the top side involves $(i, j+1 \mod N_k)$. Consequently, every τ_k-tiling can be uniquely split into blocks (macro-tiles) of size $N_k \times N_k$, where the coordinates of the cells range from $(0,0)$ in the bottom-left corner to (N_k-1, N_k-1) in the top-right corner, similarly to Fig. 2.3. Again, intuitively, each macro-tile of level k "knows" its position in the corresponding macro-tile of level $(k+1)$. For each k, the $N_k \times N_k$-macro-tile (built of tiles τ_k) should have the structure shown in Fig. 2.6, with communication wires, a computation zone, and auto-referential computation inside.

The difference from the basic construction is that now the computation simulated by a k-level macro-tile gets, as an additional input, the value k, and the zoom factor N_k is computed as a function of k. In what follows, we always assume that N_k can be computed easily given the binary expansion of k (say, in time poly$(\log N_k)$).

Technically, we assume now that the first line of the computation zone contains the following *fields* of the *input data*:

(i) the program of a Turing machine π that verifies whether a quadruple of macro-colors corresponds to a valid macro-color,

(ii) the binary expansion of the integer rank k of this macro-tile (the level in the hierarchy of macro-tiles),

(iii) the bits encoding the macro-colors: each macro-color involves the position inside the father macro-tile of rank $(k+1)$ (two coordinates modulo N_{k+1}) and $O(1)$ bits of the supplementary information assigned to the macro-colors.

Note that now the zoom factor is not provided explicitly as one of the input fields. Instead, we have the binary expansion of k, so that a Turing machine can *compute* the value of N_k, see Fig. 2.8. The difference with Fig. 2.6 is that the computation in the macro-tile of rank k gets as an input the index k instead of the universal zoom factor N.

As before, we require that the simulated computation terminates in an accepting state if the macro-colors of the macro-tile form a valid quadruple (if not, no correct tiling can be formed). The simulated computation guarantees that the macro-tiles of level k are isomorphic to the tiles of τ_{k+1}.

Note that on each level k of the hierarchy, we simulate in macro-tiles a computation of one and the same Turing machine π. Only the inputs for this machine (including the binary expansion of the rank k) vary from level to level.

This construction works well if N_k does not grow too slowly (so that the k-level macro-tiles have enough room to keep the binary expansion of k) yet not too fast (so

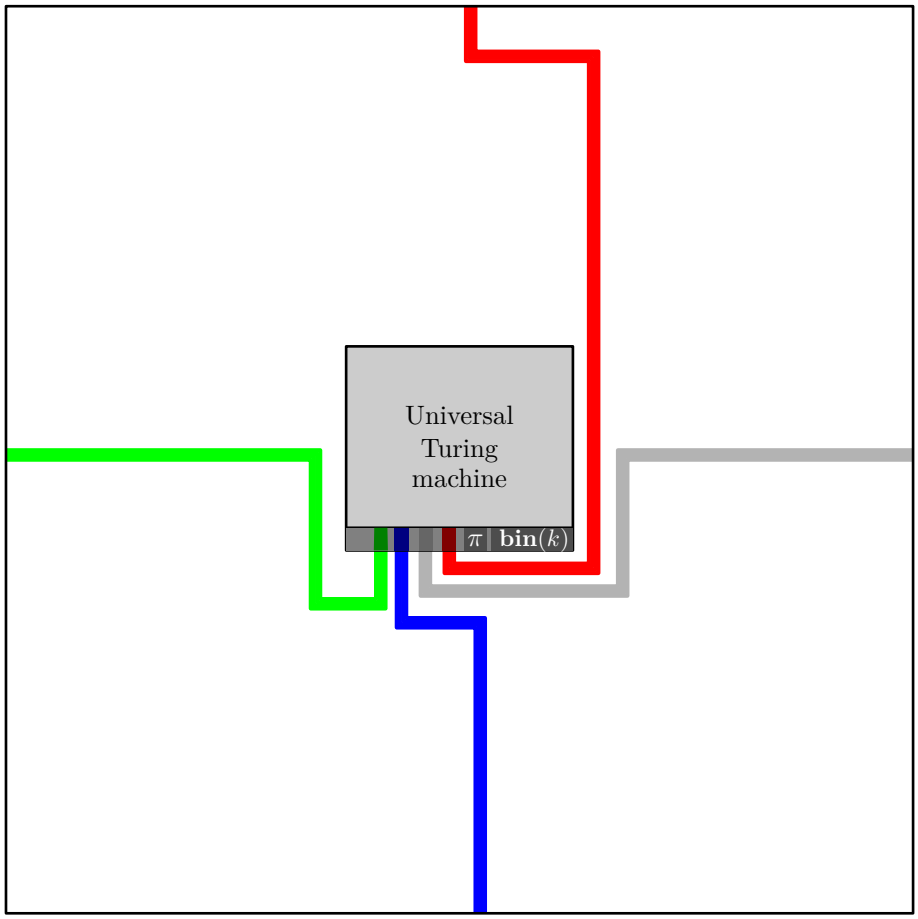

Figure 2.8: A macro-tile of level k. The computation zone represents the universal Turing machine that simulates program π, which gets as input the binary codes of the four macro-colors, the binary expansion of level k, and the text of π itself.

that the computation zone in the k-level macro-tiles can handle elementary arithmetic operations with N_{k+1}). In what follows we assume that $N_k = 3^{C^k}$ for some large enough constant C.

The growing zoom factor N_k permits to embed some *payload* in the computation zone: some "useful" computation that has nothing to do with self-simulation but affects the properties of a tiling. Since the zoom factor grows with the rank, on each subsequent level we can allocate more and more space and time to this secondary computation process.

In the next section, we show that a self-simulating tile set with variable zoom factors is useful to prove the existence of an SFT where each configuration is non-computable.

2.5.2 Embedding an infinite sequence in tiling and enforcing the non-computability of every configuration

In this section, we extend the construction discussed above and give a new proof of the following theorem.

Theorem 36 ([15], [16])**.** *For every $d > 1$ there exists a non-empty SFT on $\mathbb{Z}^d$ where all configurations are non-computable.*

In this proof we extensively use the technique of a self-simulating tiling with variable zoom factors introduced in Section 2.5.1. As mentioned above, we assume that the size of a macro-tile of rank k is equal to $N_k \times N_k$, for $N_k = 3^{C^k}$, $k = 1, 2, \ldots$, and the size of the computation zone m_k grows as $m_k = \text{poly}(\log N_k)$. We will need to superimpose another trick — a sort of embedding in a tiling of a non-computable sequence of bits.

We take as the starting point the generic scheme of a quasi-periodic self-simulating tiling explained in the previous section, and then we adjust it with some new features. From now on we require that all macro-tiles of rank k contain in their computation zone, the prefix (e.g., of length $\lceil \log k \rceil$) of some infinite sequence $X = x_0 x_1 x_2 \ldots$ We want that all macro-tiles of rank k contain one and the same prefix $x_0 x_1 x_2 \ldots x_{\lceil \log k \rceil}$, so we will embed these bits in the macro-colors of the k-level macro-tiles. To make things more pictorial, we require also that these bits be provided in the bottom line of the computation zone as one supplementary input field, as shown in Fig. 2.9.

We suppose that the computation embedded in the computational zones of the macro-tiles verifies whether or not the input data are coherent, i.e., that the bits embedded in each of the four macro-tiles match the bits $x_0 x_1 x_2 \ldots k_{\lceil \log k \rceil}$ given explicitly in this new input field. Using the usual self-simulation we can guarantee that the bits of X embedded in a macro-tile of rank $k+1$ *extend* the prefix embedded in a macro-tile of rank k. Since the size of the computation zone increases as a function of k, the entire tiling of the plane determines an infinite sequence of bits X (whose prefixes are encoded in the macro-tiles of all ranks).

Remark 20. The new feature allows *embedding* an infinite sequence X in a tiling. This embedding is *highly distributed* in the following sense: we can extract the first

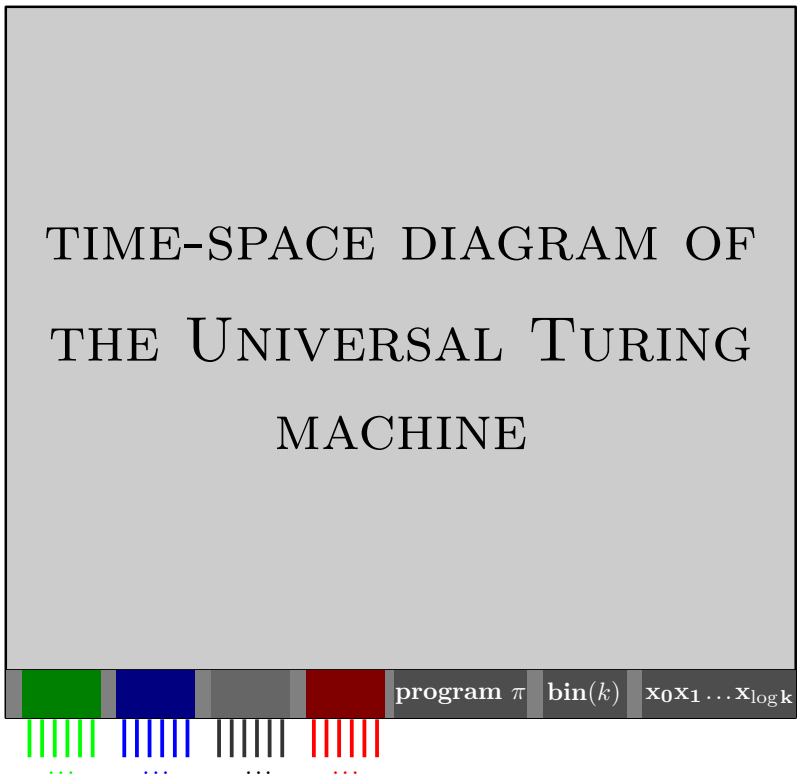

Figure 2.9: The *computation zone* of a macro-tile in the revised construction. The input data consist of the codes of the macro-colors, the simulated program π, the binary expansion of the rank k, and the first $\log k$ bits of the embedded sequence $X = x_0 x_1 x_2 \ldots$

$\log k$ bits of the sequence from every k-level macro-tile. Note that the tile set does not uniquely determine the embedded sequence X (different tilings of the same tile set can represent different sequences X). However we are going to control the class of sequences X that can be embedded in a tiling.

When the class of embedded sequences of X is not restricted, this construction is of no interest. It becomes meaningful if we can enforce some special properties of the embedded sequence X. Let us show that we can guarantee that for all valid tilings the embedded sequence X is *not computable*. To achieve this property, we make the machine in the computation zone enumerate two non-separable enumerable sets (on each level k we run these two enumerations for the number of steps that fits the computation zone available in a macro-tile of rank k). Then, let us require that X is a separator between these two sets. Technically, on each level of the hierarchy the computation in each macro-tile verifies that these (partially) enumerated sets are indeed separated by the given prefix of X. Such a sequence X must be non-computable. Combining all elements together, we obtain a tile set τ that enjoys two nontrivial properties: all τ-tilings are non-computable and quasiperiodic. Thus, we have proved Theorem 36.

2.6 Quasiperiodic self-simulating SFT

In this section, we revise once again the construction of a self-simulating tiling and enforce the property of quasi-periodicity or even minimality. In particular, this construction will provide a new proof of Theorem 37. To implement this construction, we have to superimpose some new properties of a self-simulating tiling.

2.6.1 Preliminary remarks: Supplementary constraints that can be imposed on a self-simulating tiling

The tiles involved in our self-simulating tiles set (as well as all macro-tiles of each rank) can be classified into three types:

(a) "skeleton" tiles, which keep no information except for their coordinates in the father macro-tile (the white area in Fig. 2.8; each of these tiles looks like the tile shown in Fig. 2.3): these tiles work as building blocks of the hierarchical structure;

(b) "communication wires," which transmit the bits of the macro-colors from the borderline of the macro-tile to the computation zone (the colored lines in Fig. 2.8; each of these tiles looks like the tiles shown in Fig. 2.4);

(c) tiles of the computation zone (intended to simulate the space-time diagram of the Universal Turing machine, the gray area in Fig. 2.8).

Each pattern that includes only "skeleton" tiles (or "skeleton" macro-tiles of some rank k) reappears infinitely often in all homologous positions inside all macro-tiles of higher rank. Unfortunately, this property is not true for those patterns that involve

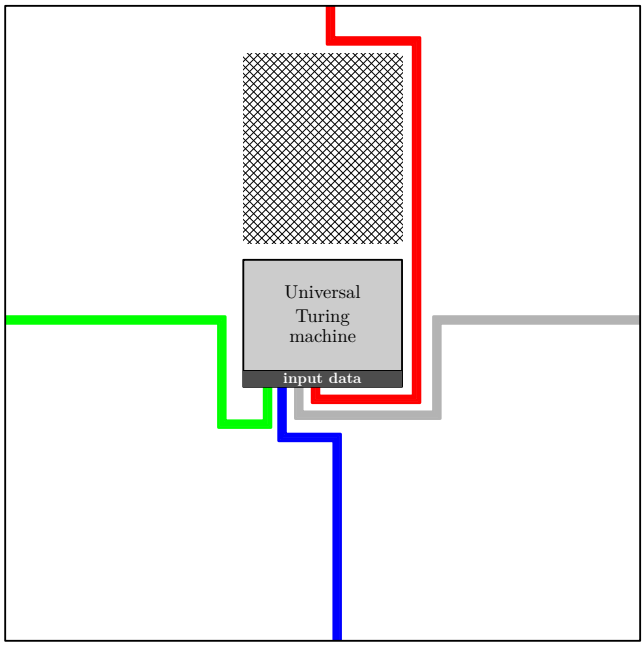

Figure 2.10: The "free" area reserved above the computation zone.

the "communication zone" or the "communication wires." Thus, the basic construction of a self-simulating tiling does not imply the property of quasiperiodicity. To overcome this obstacle, we need several new technical tricks.

First of all, we impose several restrictions on our construction of a self-simulating tiling. These restrictions in themselves do not make the tilings quasiperiodic, but they do simplify the upcoming revision of the construction. More specifically, we enforce the following additional properties (p1)–(p4) of a tiling, with only a minor modification of the construction.

(p1) In our basic construction, each macro-tile contains a computation zone of size m_k, which is *much* less than the size of the macro-tile N_k. In what follows, we need to reserve free space in a macro-tile, in order to insert $O(1)$ (some constant number) copies of each 2×2 pattern from the computation zone (of this macro-tile), right above the computation zone. This requirement is easy to meet. We assume that the size of a k-level macro-tile (measured in blocks that are themselves macro-tiles of level $k-1$) is $N_k \times N_k$, and the computation zone in this macro-tile is $m_k \times m_k$ for $m_k = \text{poly}(\log N_k)$. Therefore, we can reserve an area of size $\Theta(m_k)$ right above the computation zone, which is free of "communication wires" or any other functional gadgets, see the "empty" hatched area in Fig. 2.10. So far, this area consisted of only skeleton tiles; in what follows (Section 2.6.2 below), we will use this zone to place some new nontrivial elements of the construction.

(p2) We require that the tiling inside the computation zone satisfies the property of 2×2-*determinacy*. That is, if we know all of the colors on the borderline of a 2×2-pattern inside the computation zone (i.e., a tuple of 8 colors), then we can reconstruct the four tiles of this pattern. Again, we do not need any new ideas to implement this property. It is not hard to see that this requirement is met if we represent the space-time diagram of a Turing machine in a natural way (see the discussion on p. 110).

(p3) The communication channels in a macro-tile (the wires that transmit information from the macro-color on the borderline of this macro-tile to the bottom line of its computation zone) must be isolated from each other. The distance between every two wires must be greater than two cells, as shown in Fig. 2.11. In other words, each group of cells of size 2×2 can touch at most one communication wire. Since the number of wires in a k-level macro-tile is only $O(\log N_k)$, we have enough free space to lay the "communication cables" maintaining the required safety gap, so this constraint is easy to satisfy.

(p4) In our construction, the macro-colors of a k-level macro-tile are encoded by bit strings of length $r_k = O(\log N_{k+1})$. In the previous section, we only assumed that this encoding was somewhat "natural" and easy to handle. So far, the choice of encoding has been of little importance: we have only required that some natural manipulations with macro-colors could be implemented in polynomial time.

We now add another (seemingly artificial) condition. We have decided that each macro-color is encoded in a string of r_k bits. We require now that each bit in this encoding takes both values 0 and 1 *quite often*. More precisely, we require that for each $i = 1, \ldots, r_k$ there are *quite many* macro-tiles where the ith bit of encoding of the top (bottom, left, right) macro-color is equal to 0, and there are *quite many* other macro-tiles where the ith bit of this encoding is equal to 1. In what follows we specify what the words *quite often* and *quite many* mean in this context.

Technically, we use the following property: for every position $s = 1, \ldots, r_k$ and for every $i = 0, \ldots, N_{k+1} - 1$ we require that

- there exists j_0, such that the sth bit in the top, left, and right macro-colors of the k-level macro-tile at the positions (i, j_0) in the $(k+1)$-level father macro-tile are equal to 0, and
- there exists j_1, such that the sth bit in the top, left, and right macro-colors of the k-level macro-tile at the positions (i, j_1) in the $(k+1)$-level father macro-tile are equal to 1.

There are many (more or less artificial) ways to implement this constraint. For example, we may subdivide the array of r_k bits encoding a macro-color into three equal zones of size $r_k/3$ and require that for each macro-tile only one of these three zones contains the "meaningful" bits, and the two other zones contain only zeros and ones respectively; we require then that the "roles" of these three zones cyclically permute as we go upwards along a column of macro-tiles, see Fig. 2.12.

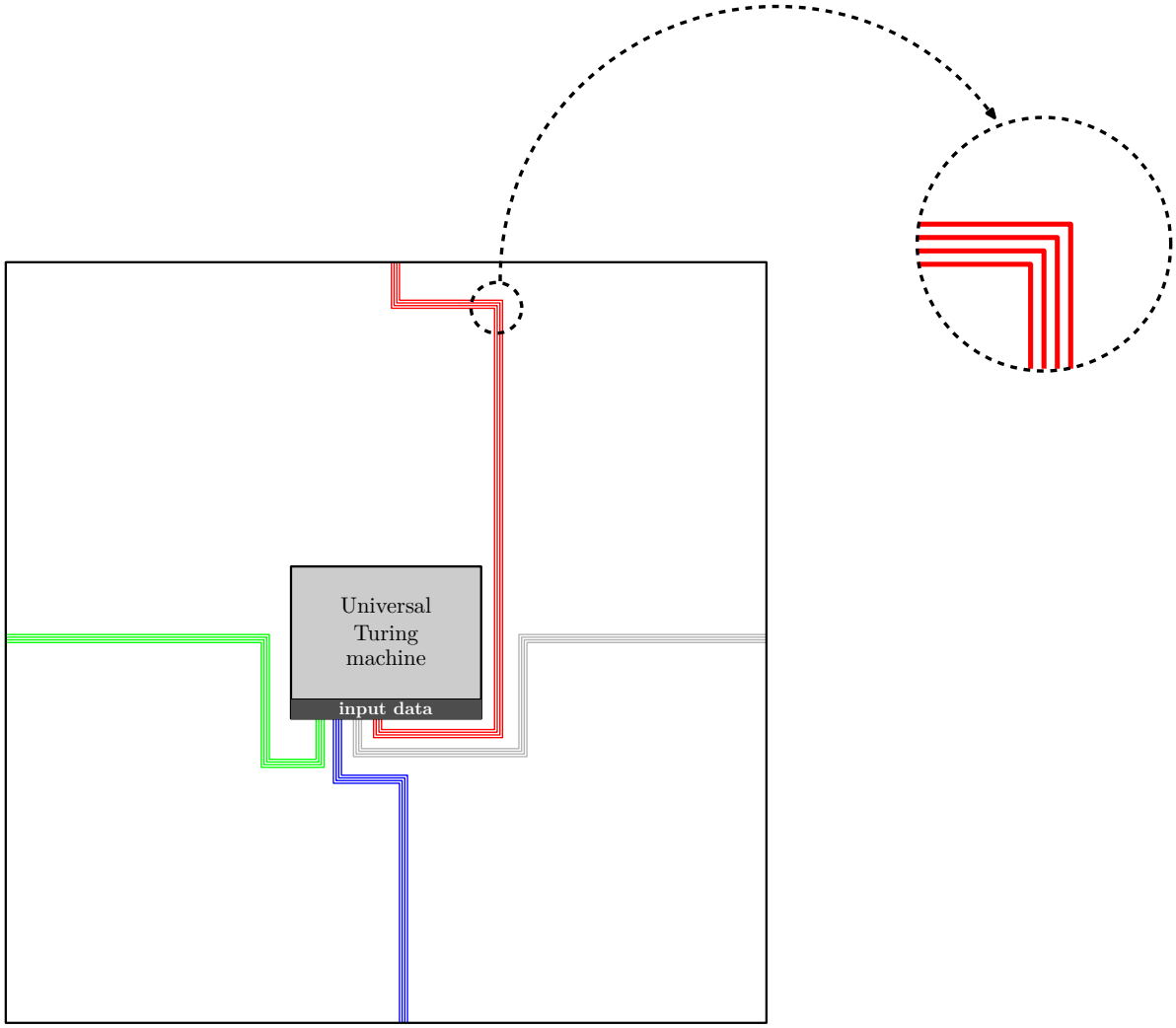

Figure 2.11: The wires in the "communication cables" (shown in red, blue, green, and gray in the figure) are separated by a gap (shown in white), so that the distance (measured in tiles) between every two wires is greater than two.

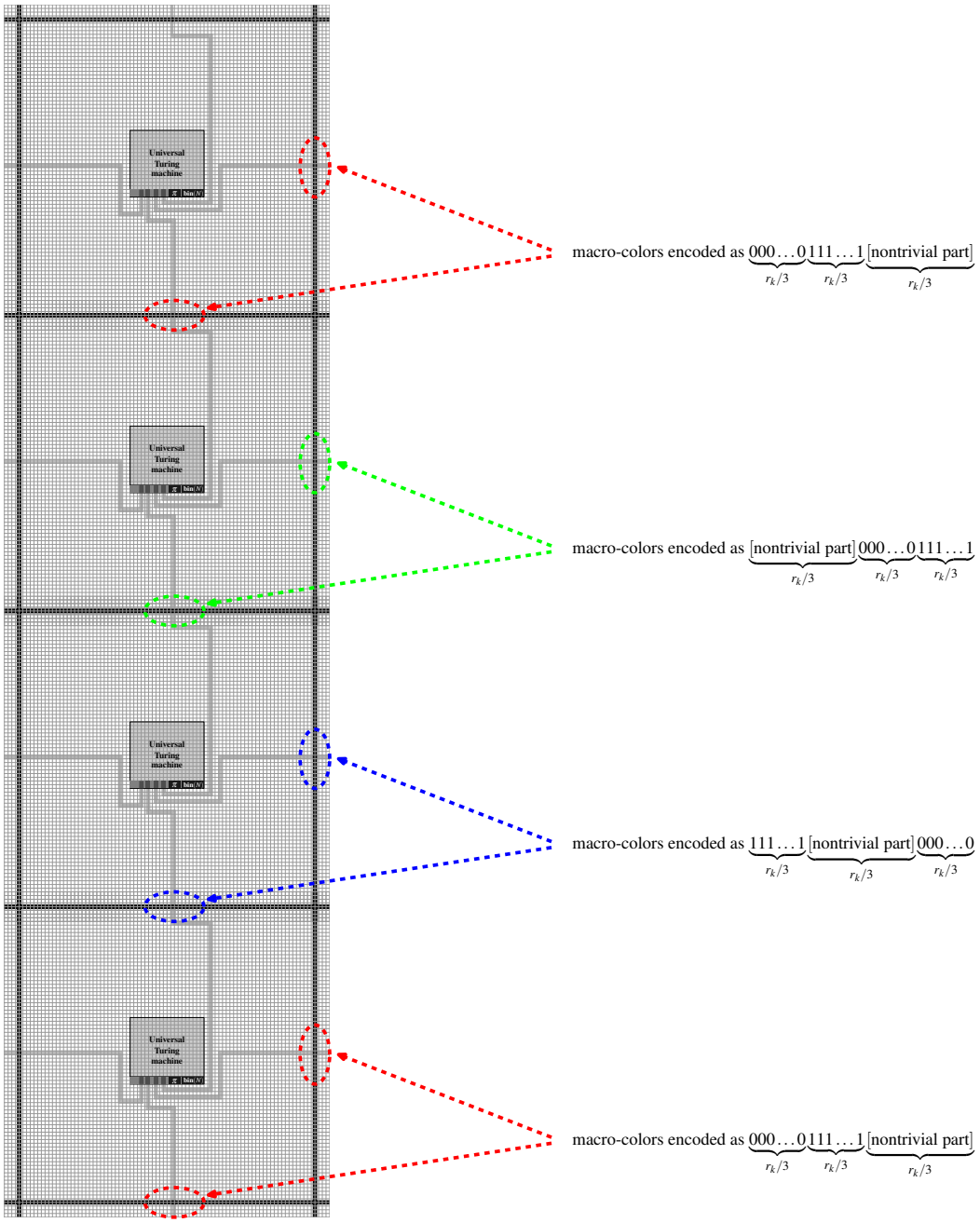

Figure 2.12: Encoding of macro-colors. The r_k bits of the code are split into three blocks of size $r_k/3$. One of them consists of all 0s, another of all 1s, and only the third one contains a nontrivial binary code. The roles of the three blocks change cyclically from one macro-tile to another when we move upwards.

2.6.2 Aperiodicity and minimality

To achieve the property of minimality of an SFT, we should guarantee that every finite pattern that appears *once* in at least one tiling must also appear in *every large enough square* in every tiling. In a tiling with a hierarchical structure of macro-tiles each finite pattern can be covered by at most four macro-tiles (by a 2×2-pattern) of an appropriate rank. Hence, to guarantee minimality, it is enough to show that every 2×2-block of macro-tiles of any rank k that appears in at least one τ-tiling actually reappears in this tiling in every large enough square. Let us classify all 2×2-block of macro-tiles (by their position in the father macro-tiles of higher rank) and discuss what revisions of the construction are required.

Case 1: Skeleton tiles. For a 2×2-block of four "skeleton" macro-tiles of level k, there is nothing to do. Indeed, in our construction we have exactly the same blocks with every vertical shift by a multiple of L_{k+1} (we have there a similar block of k-level "skeleton" macro-tiles within another macro-tile of rank $(k+1)$).

Case 2: Communication wires. Let us consider the case when a 2×2-block of k-level macro-tiles involves a part of a communication wire. Due to property (p3) we may assume that only one wire is involved. The bit transmitted by this wire is either 0 or 1; in either case, due to property (p4), we can find another similar 2×2-block of k-level macro-tiles (at the same position within the father macro-tile of rank $(k+1)$ and with the same bit included in the communication wire) in every macro-tile of level $(k+2)$. In this case we can find a duplicate of the given block with a vertical shift of size $O(L_{k+2})$.

Case 3: Computation zone. We now consider the most difficult case: when a 2×2-block of k-level macro-tiles touches the computation zone. In this case we cannot obtain the property of quasiperiodicity for free, and so we have to make one more modification of our general construction of a self-simulating tiling.

Note that for each 2×2-window that touches the computation zone of a macro-tile, there are only $O(1)$ ways to tile them correctly. For each possible position of a 2×2-window in the computation zone, and for each possible filling of this window by tiles, we reserve a special 2×2-*slot* in a macro-tile, which is essentially a block of size 2×2 in the "free" zone of a macro-tile. We refer to this gadget as a *diversification slot*. These slots will enforce the property of "diversity": for every small pattern that *could* appear in the computation zone, we will guarantee that it *must* appear in the corresponding diversity slot in every macro-tile of the same rank.

These diversification slots must be placed far away from the computation zone and from all communication wires. We prefer to place every diversification slot in the same vertical stripe as the "original" position of this block, as shown in Fig. 2.13 (this property of vertical alignment will be used in Section 2.8). We have enough free space to place all necessary diversification slots, due to property (p1). We define the neighbors around each diversification slot in such a way that only one specific 2×2 pattern can patch it (here we use the property (p2)).

In our construction, the tiles around this slot "know" their real coordinates in the bigger macro-tile, while the tiles inside the diversification slot do not (they "believe" they are tiles in the computation zone, while in fact they belong to an artificial isolated "diversity preserving" slot far outside of any real computation), see Fig. 2.13

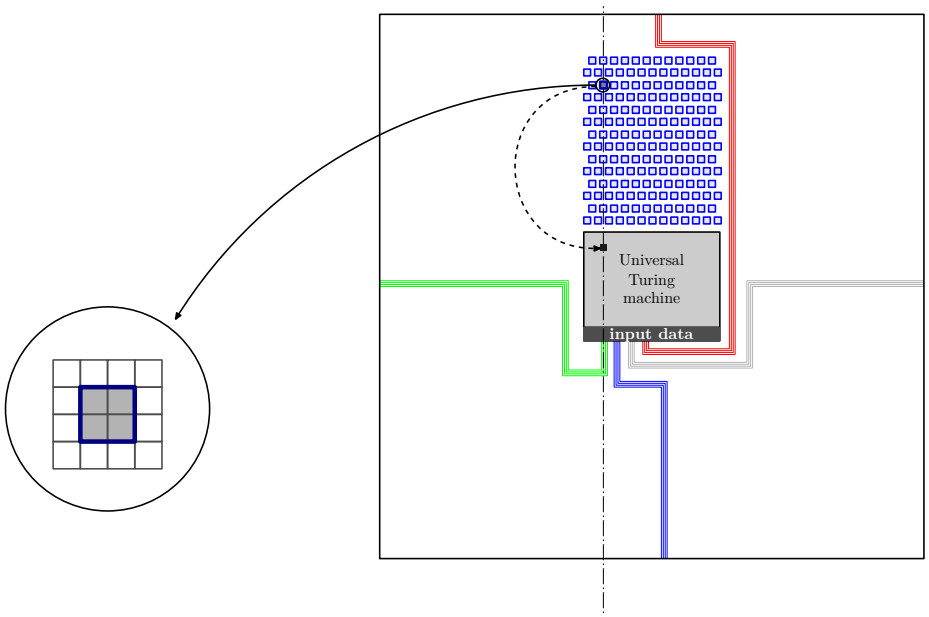

Figure 2.13: Positions of the *diversification slots* for patterns from the computation zone.

and Fig. 2.14. The frame of the diversification slot consists of 12 "skeleton" tiles (the white squares in Fig. 2.14); they form a slot that involves inside a 2×2-pattern extracted from the computation zone (the gray squares in Fig. 2.14). In the picture, we show the "coordinates" encoded in the colors on the sides of each tile. Note that the colors of the bold lines (the blue lines between the white and gray tiles and the bold black lines between the gray tiles) should contain some information beyond the coordinates—these colors involve the bits used to simulate a space-time diagram of the universal Turing machine. In this picture, the "real" coordinates of the bottom-left corner of this slot are $(i+1, j+1)$, while the "natural" coordinates of the pattern inside the diversification slot (when this pattern appears in the computation zone) are (s, t).

We choose the positions of the diversification slots in the macro-tile so that the coordinates can be computed by some simple algorithm in time polynomial in $\log N_k$. We require that all diversification slots be detached from each other in space, so they do not damage the general structure of the "skeleton" tiles building the macro-tiles.

Now it is not hard to see that for the revised tile set, every pattern that appears *at least once* in *at least one* tiling must in fact appear in *every large enough* pattern in *every* tiling. Thus, the revised construction of a self-simulating tiling guarantees that

- every tiling is aperiodic (the argument from the previous section remains

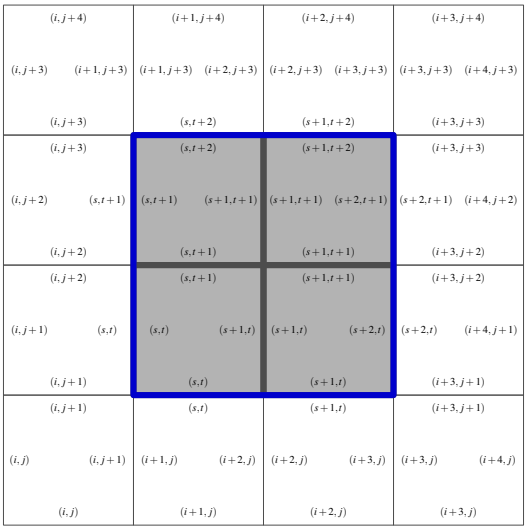

Figure 2.14: A diversification slot for a 2×2-pattern from the computation zone.

valid), and

- every pattern that appears at least once in at least one configuration must appear in every large enough square in every tiling.

Thus, our construction implies the following theorem.

Theorem 37 (Bailler–Ollinger)**.** *For every $d > 1$ there exists a non-empty SFT on $\mathbb{Z}^d$ where each configuration is aperiodic and quasiperiodic* (*and even minimal*).

(This result was originally proven in [78] with a different technique, for a tile set τ constructed in [77].)

2.7 Quasiperiodicity and non-computability

In this section, we construct an SFT in which all configurations are simultaneously quasiperiodic and non-computable. To this end we merge together the constructions from Section 2.5.2 and Section 2.6. We start with a self-simulating tile set with a variable zoom factor and an embedded sequence X, as explained in Section 2.5.2 (in what follows we will restrict the class of the embeddable sequences). Then we superimpose the features from Section 2.6, which imply quasiperiodicity.

Since the sequence X embedded in a tiling is not uniquely defined by the tile set, different τ-tilings involve different embedded sequences X and therefore different finite patterns. Thus, the defined SFT is not minimal. However, it has the property of *quasiperiodicity*. Indeed, every valid tiling contains a well-defined embedded sequence X. Let us fix one infinite sequence X and restrict ourselves to the class of tilings $\mathcal{T}(X)$ that represent this specific sequence. Then, all k-level macro-tiles in

every tiling in $\mathcal{T}(X)$ involve (one and the same) prefix of X. Now the argument from Section 2.6, repeated word for word gives the following property.

Every 2×2-block of k-level macro-tiles that appears at least once in at least one tiling in $\mathcal{T}(X)$, must reappear in every large enough pattern of every tiling in $\mathcal{T}(X)$.

Hence, all implied tilings are quasiperiodic.

This construction becomes more interesting if we restrict the class of sequences X that can be embedded in a tiling. Similarly to the proof of Theorem 36 in Section 2.5.2, we require that the computation embedded in each macro-tile enumerates some set of "forbidden factors" and verifies that the embedded X contains non of these factors. The choice of the (recursively enumerable) class of factors "forbidden" for the embedded X will determine the properties of the tiling.

We start with the property of non-computability. As we explained in Section 2.5.2, we can choose the set of forbidden factors so that the embedded sequence X must be *not computable*. This gives the following result.

Theorem 38. *For every $d > 1$ there exists a non-empty SFT on $\mathbb{Z}^d$, where all configurations are non-computable and quasiperiodic.*

Thus, we have constructed a tile set τ, such that all τ-tilings are quasiperiodic and non-computable. Up to now, we could not say much more about the degree of unsolvability (Turing degrees) of the implied τ-tilings. Now will we enhance this construction by implementing some more precise control over the class of embeddable sequences X, and therefore over the class of possible Turing degrees of τ-tilings. We start with a proposition that characterizes the no-go zones for this technique.

Theorem 39. *(a) Every SFT is effectively closed. (b) For every infinite minimal SFT $\mathcal{S}$, the class of the Turing degrees representable by configurations in $\mathcal{S}$ is upper-closed: if there exists a τ-tiling that has a Turing degree T, then every Turing degree $T' > T$ is also represented by some τ-tiling.*

Proof. (a) is trivial, (b) is proven in [89]. □

Remark 21. Observe that we cannot guarantee that the Turing degrees of *all* τ-tilings are *very* high. More specifically, we cannot guarantee that all τ-tilings are *not low* or *not hyperimmune-free*. Indeed, due to the low basis theorems, for every tile set τ, some τ-tilings are not low and not hyperimmune-free.

The next theorem essentially claims that a class of Turing degrees which is not forbidden by Theorem 39 (i.e., a class that is upwards-closed and corresponds to an effectively closed set) can be implemented by a suitable tile set.

Theorem 40. *For every effectively closed set $\mathcal{A}$, and for every $d > 1$, there exists a non-empty SFT $\mathcal{S}$ in $\mathbb{Z}^d$, in which all configurations are quasiperiodic, and the Turing degrees of all configurations in $\mathcal{S}$ form exactly the upper closure of $\mathcal{A}$ (defined as the set of all Turing degrees d, such that $d \geq_T \omega$ for at least one $\omega \in \mathcal{A}$).*

Proof of Theorem 40. To prove this theorem, we again employ the idea of embedding of an infinite sequence X in a tiling, and control more precisely the properties of the embedded sequence. Similarly to the construction discussed above, we require that all macro-tiles of rank k involve the same finite sequence of $\log k$ bits on their computation zone, which is understood as a prefix of X. We can guarantee that the prefix embedded in macro-tiles of rank k is compatible with the prefix available to the macro-tiles of the next rank $(k+1)$.

Further, since $\mathscr{A}$ is in Π^0_1, we can enumerate the (potentially infinite) list of patterns that should not appear in X. On each level, the macro-tiles run this enumeration for the available space and time (limited by the size of the computation zone available on this level), and verify that the discovered forbidden patterns do not appear in the prefix of X accessible to the macro-tiles of this level. Since the computation zone becomes increasingly bigger with each level, the enumeration extends longer and longer. Thus, a sequence X can be embedded in an infinite tiling, if and only if, this sequence does not contain any forbidden pattern (i.e., if this X belongs to $\mathscr{A}$).

What are the Turing degrees of the tilings in the described tile set? In our tile set, every tiling is defined uniquely by the following information: the sequence X embedded in this tiling, and the sequences of integers σ_h, σ_v that specify the shifts (the vertical and the horizontal ones) of macro-tiles of each level relative to the origin of the plane. Indeed, on each level k we split the macro-tiles of the previous rank into blocks of size $N_k \times N_k$. These blocks make k-level macro-tiles, and there are N_k^2 ways to choose the grid of horizontal and vertical lines that define this split. Given the sequences σ_h, σ_v and an $X \in \mathscr{A}$, we can reconstruct the entire tiling. It remains to note that σ_h and σ_v can be absolutely arbitrary. Thus, the Turing degree of a tiling is the Turing degree of (X, σ_h, σ_v), which can be an arbitrary degree not less than X. That is, the set of degrees of tilings is exactly the closure of $\mathscr{A}$, i.e., the set of all Ys that are not less than some $X \in \mathscr{A}$. So we get the statement of Theorem 40. □

2.8 On the subdynamics of co-dimension 1 for self-simulating SFTs

In Section 2.5.2 and Section 2.7, we used a sort of embedding of one-dimensional sequences in a two-dimensional SFT. That embedding was highly distributed: the first $\log k$ bits of the sequence embedded in a configuration could be found in every macro-tile of rank k of this configuration. In this section we discuss a different way of embedding a (bi-infinite) one-dimensional sequence in two-dimensional configurations of an SFT; this version of embedding is less distributed and more local. With this technique we will be able to control the subdynamics of a two-dimensional shift, and as a result we will prove Theorems 41, 44, 45.

In this section we prove the following theorem:

Theorem 41. *(a) Let $\mathscr{A}$ be an effective $\mathbb{Z}^d$-shift over an alphabet Σ_A. Then there exists a quasiperiodic SFT $\mathscr{B}$ (over another alphabet Σ_B) of dimension $d+1$ such that $\mathscr{A}$ is* simulated *by $\mathscr{B}$ in the sense of Definition 13.*

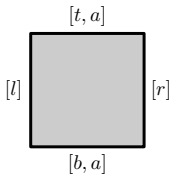

Figure 2.15: A tile propagating a letter $a \in \Sigma$ in the vertical direction. Formally speaking, this tile is a quadruple of *colors*, the left side has color $[l]$, the right side has color $[r]$, the top and the bottom sides have colors $[t,a]$ and $[b,a]$, respectively. The colors for the top and bottom sides involve a letter from Σ. We allow only tiles where the colors of the top and bottom sides involve one and the same letter.

(b) Let $\mathscr{A}$ be a $\mathbb{Z}^d$-shift simulated in the sense of Definition 13 by some SFT $\mathscr{B}$ of dimension $d+1$. Then $\mathscr{A}$ is effectively closed.

Statement (b) of the theorem is simple, so we focus on statement (a). We now embed a bi-infinite sequence $\mathbf{x} = (x_i)$ over an alphabet Σ in our tiling. To this end, we assume that each individual τ-tile "keeps in mind" a letter from Σ that propagates without change in the vertical direction. Formally speaking, a letter from Σ should be a part of the top and bottom colors of every τ-tile (the letters assigned to both sides of a tile must be equal to each other), see Fig. 2.15 and Fig. 2.2. We want to guarantee that a Σ-sequence can be embedded in a τ-tiling, if and only if, this sequence belongs to a fixed given effective shift $\mathscr{A}$.

The general plan is to "delegate" the factors of the embedded sequence into the computation zones of macro-tiles, where these factors can be validated (that is, the simulated Turing machine can verify whether or not these factors contain any forbidden subwords). By using tilings with growing zoom factors, we can guarantee that the size of the computation zone of a k-rank macro-tile grows with k. So we have at our disposal the computational resources required to run all necessary validation tests on the embedded sequence. It remains to organize the propagation of the letters from the embedded sequence to the "conscious memory" (the computation zones) of the macro-tiles of all ranks. In what follows we explain how this propagation is organized.

The zone of responsibility of a macro-tile. In our construction, a macro-tile of level k is a square of size $L_k \times L_k$, with $L_k = N_1 \cdot N_2 \cdot \ldots \cdot N_k$ (where N_i is the zoom factor on level i of the hierarchy of macro-tiles). We say that a k-level macro-tile is *responsible* for the letters of the embedded sequence $\mathbf{x}$ assigned to the columns of the (ground level) tiles of this macro-tile as well as to the columns of macro-tiles of the same rank on its left and on its right. That is, the *zone of responsibility* of a k-level macro-tile is a factor of length $3L_k$ from the embedded sequence, see Fig. 2.16. (The zones of responsibility of two vertically aligned macro-tiles are the same; the zones of responsibility of two horizontally neighboring macro-tiles overlap.)

Letter assignment: The computation zone of a k-level macro-tile (of size $m_k \times m_k$) is too small to contain all the letters from its zone of responsibility. So we require

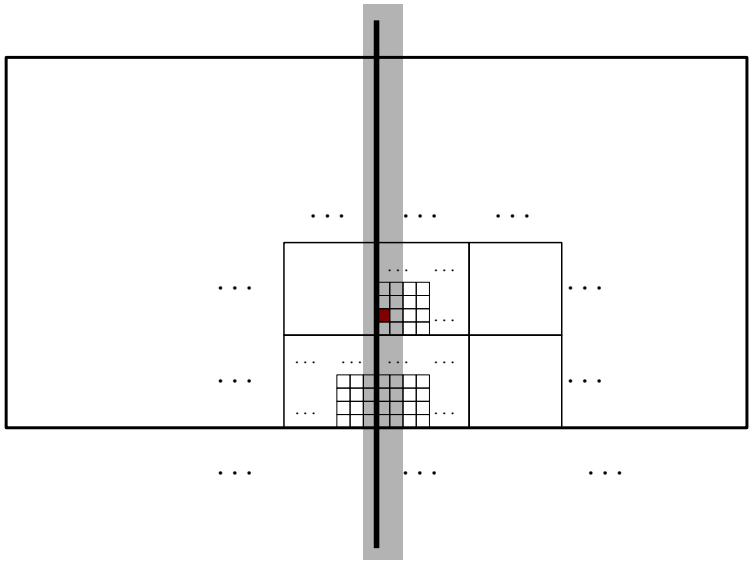

Figure 2.16: The zone of responsibility (the gray vertical stripe) of a macro-tile (the red square) is three times wider than the macro-tile itself.

that the computation zone obtains as an input a (short enough) chunk of letters from its zone of responsibility. Let us say that it is a factor of length $l_k := \log\log L_k$ from the stripe of $3L_k$ columns constituting the zone of responsibility of this macro-tile. We will say that this chunk is *assigned* to this macro-tile.

The infinite stripe of vertically aligned k-level macro-tiles shares the same zone of responsibility. However, different macro-tiles in such a stripe will obtain different assigned chunks. The choice of the assigned chunk varies from 0 to $(3L_k - l_k)$. Therefore, we need to choose for each k-level macro-tile a position of a factor of length l_k in its zone of responsibility of length $3L_k$. This choice is quite arbitrary. Let us say, for definiteness, that for a macro-tile M of rank k the first position of the assigned chunk (in the stripe of length $3L_k$) is defined as the vertical position of M in the father macro-tile of rank $(k+1)$ (taken modulo $(3L_k - l_k)$).

Remark 22. We have chosen zoom factors N_k growing doubly exponentially in k, so $N_{k+1} \gg 3L_k$. Hence, every chunk of length l_k from a stripe of width $3L_k$ is assigned to some (actually, to infinitely many) of the macro-tiles "responsible" for these $3L_k$ letters.

Remark 23. Since the zones of responsibility of neighboring k-level macro-tiles overlap by more than l_k, every finite factor of length l_k in the embedded sequence $\mathbf{x}$ is assigned to some k-level macro-tile (even if it does not sit in one macro-tile and involves columns of two neighboring macro-tiles of level k), see Fig. 2.17.

Implementing the letter assignment by self-simulation. In the *letter assignment* paragraph above, we formulated several requirements: how the data should be

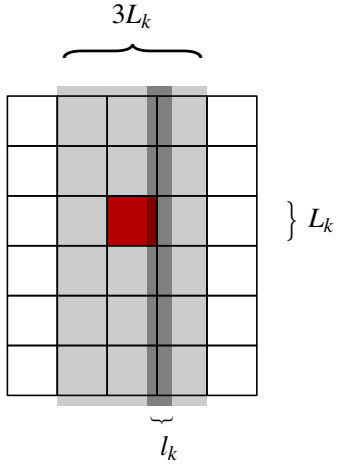

Figure 2.17: The macro-tile of size L_k (shown in red) is *responsible* for the vertical stripe of width $3L_k$ shown in light-gray (three times wider than the macro-tile itself). Such a macro-tile can handle a factor of length l_k of the embedded sequence that corresponds to the group of columns that touch this macro-tile as well as its neighbor on the right or on the left (an example is shown in dark-gray).

propagated from the ground level (individual tiles) to k-level macro-tiles. That is, for each k-level macro-tile $\mathscr{M}$ we specified which chunk of the embedded sequence should be a part of the data fields in the computation zone of $\mathscr{M}$. So far we have not explained how this propagation can be implemented, i.e., how the assigned chunks can arrive at the high-level data fields. Now, we will explain how to implement the required scheme of letter assignment in a self-simulating tiling. Technically, we append to the input data of the computation zones of macro-tiles some supplementary data fields:

(iv) the block of l_k letters from the embedded sequence assigned to this macro-tile,

(v) three blocks of bits of l_{k+1} letters of the embedded sequence assigned to this father macro-tile, and two uncle macro-tiles (the left and the right neighbors of the father),

(vi) the coordinates of the father macro-tile in the "grandfather" (of rank $(k+2)$).

(I.e., the first line of the computation zone still looks similar to Fig. 2.9, but now it contains more input data in the first line of the computation zone). Speaking informally, the computation in each k-level macro-tile must check the consistency of the data in fields (iv), (v), and (vi). That is, if some letters from the fields (iv) and (v) correspond to the same vertical column in the zone of responsibility, then these letters must be equal to each other. Also, if a k-level macro-tile plays the role of a cell in the computation zone of the $(k+1)$-level father, it should check the consistency of its

(v) and (vi) with the bits displayed in the father's computation zone. Lastly, we must ensure the coherence of the fields (v) and (vi) for each pair of neighboring k-level macro-tiles; so this data should also be a part of the macro-colors.

Note that the data from the uncles' macro-tiles are necessary to deal with letters from the columns that physically belong to the neighboring macro-tiles. So the consistency of the fields (v) is imposed also on neighboring k-level macro-tiles that belong to different $(k+1)$-level fathers (the borderline between these k-level macro-tiles is also the borderline between their fathers).

The computations verifying the coherence of the new fields can be performed in polynomial time, and the required update of the construction fits the general constraints on the parameter. (See also the discussion on "letter delegation" in [j7, Section 7]).

Concluding remarks: Testing against forbidden factors. To guarantee that the embedded sequence $\mathbf{x}$ contains no forbidden patterns, each k-level macro-tile should allocate some part of its computation zone to enumerating (within the limits of available space and time) the forbidden pattern, and verify that the block of l_k letters assigned to this macro-tile contains none of the forbidden factors found.

The time and space allocated to enumerating the forbidden words grow as functions of k. To ensure that the embedded sequence contains no forbidden patterns, it is enough to guarantee that each forbidden pattern is found by macro-tiles of high enough rank, and every factor in the embedded sequence is compared (on some level of the hierarchy) with every forbidden factor.

Thus, we get a construction of a two-dimensional tiling that simulates a given effective one-dimensional shift: for a given effective one-dimensional shift $\mathscr{A}$, we can construct a tile set τ such that the bi-infinite sequences that can be embedded in τ-tilings are exactly the sequences of $\mathscr{A}$, which is the statement of Theorem 41 (a).

In Section 2.10, we explain how to make these tilings quasiperiodic when the simulated one-dimensional shift is also quasiperiodic.

2.9 High Kolmogorov complexity and strongly aperiodic SFTs

In this section, we prove two strengthenings of Theorem 35: we show that some $\mathbb{Z}$-SFTs contain only *strongly aperiodic* configurations (for such a configuration, every translation must change *many* letters), and some $\mathbb{Z}$-SFTs contain only patterns of high Kolmogorov complexity. We start with the latter statement.

Theorem 42 ([75])**.** *There exist a tile set τ and constants $c_1 > 0$ and c_2, such that τ-tilings exist, and in every τ-tiling T every $N \times N$ square has a Kolmogorov complexity of at least $c_1 N - c_2$.*

Proof. We say that a bi-infinite sequence of bits $\mathbf{x} = (x_i)$ has *Levin's property* if every N-bit factor w of $\mathbf{x}$ has complexity $\Omega(N)$. More precisely, there exist constants $c_1, c_2 > 0$, such that for all factors w of $\mathbf{x}$

$$C(w) \geq c_1 |w| - c_2.$$

Such a bi-infinite sequence exists (see Lemma 1). The class $\mathscr{L}_{c_1,c_2}$ of all these sequences is an effective $\mathbb{Z}$-shift, since there exists an algorithm that enumerates the set of all forbidden patterns (i.e., all words w such that $C(w) < c_1|w| - c_2$).

Consequently, we can apply Theorem 41 (a) and embed $\mathscr{L}_{c_1,c_2}$ in a $\mathbb{Z}^2$-tiling. For the resulting tiling, for every $(n \times n)$-pattern we can effectively reconstruct the n bits assigned to the columns that intersect this pattern. Hence, the Kolmogorov complexity of every $(n \times n)$-pattern is at least $\Omega(n)$, and we are done. □

Remark 24. Theorem 42 can be strengthened: not only for some $c_1 > 0$ but also for *every* $c_1 > 0$ there exists a number c_2 and a tile set τ, such that τ-tilings exist, and in every τ-tiling T every $N \times N$ square has a Kolmogorov complexity of at least $c_1 N - c_2$. To prove this version of the theorem we need to embed in a tiling the sequences $\mathbf{x}$ with Levin's property over a large enough alphabet.

Remark 25. A standalone pattern of size $n \times n$ over an alphabet Σ (with at least two letters) can have a Kolmogorov complexity up to $\Theta(n^2)$. However, this density of information cannot be enforced by local rules, because in every SFT on $\mathbb{Z}^2$ there exists a configuration such that the Kolmogorov complexity of all $N \times N$-patterns is bounded by $O(n)$, see [75]. Thus, the lower bound $\Omega(n)$ in Theorem 42 is optimal in the class of all $\mathbb{Z}^2$-SFTs.

Definition 14. *Let $\alpha > 0$ be a real number and Σ be a finite alphabet. We say that a configuration $\mathbf{f}\colon \mathbb{Z}^2 \to \Sigma$ is* α-aperiodic *if for every non-zero vector $(u, v) \in \mathbb{Z}^2$, there exists N, such that in every square whose side is at least N, the fraction of points with coordinates (x, y), such that $\mathbf{f}(x, y) \neq \mathbf{f}(x + u, y + v)$, is at least α.*

Remark 26. Every α-aperiodic configuration is far from every periodic configuration. More precisely, if a configuration $\mathbf{x}$ is α-aperiodic, then the Besicovitch distance between $\mathbf{x}$ and any periodic configuration $\mathbf{y}$ is at least $\alpha/2$. (The Besicovitch distance between two configurations is defined as

$$\limsup_N \operatorname{dens}_N(\mathbf{x}, \mathbf{y}),$$

where dens_N is the fraction of points where two configurations differ in the $N \times N$ centered square. It is easy to see that the Besicovitch distance between every two configurations is shift-invariant, i.e., does not depend on the choice of the central point.)

Theorem 43. *There exists an $\alpha > 0$ and a tile set τ, such that τ-tilings exist and every τ-tiling is α-aperiodic.*

Proof. We can deduce this result from Theorem 42. Indeed, due to Theorem 42 we can construct a tile set such that every tiling embeds a horizontal sequence with high-complexity substrings, and such a sequence cannot match itself well after a shift. (If the embedded sequence is binary, we can guarantee that every shift changes about 50% of its bits, e.g., between 49% and 51% of bits in every large enough factor.)

We take the direct product of this tile set with its own copy rotated by 90°. Then, any non-zero translation of a tiling will shift either the vertical or the horizontal embedded sequence, and therefore change a constant fraction of the positions. □

Remark 27. The construction above works with every constant $\alpha < 1$. We only need to apply a stronger version of Theorem 42 (with embedded Levin's sequences with high enough complexity of factors).

2.10 Subdynamics of quasiperiodic shifts

Theorem 41 can be transposed to the classes of quasiperiodic and minimal shifts:

Theorem 44. *(a) Let $\mathscr{A}$ be an effective quasiperiodic $\mathbb{Z}^d$-shift over some alphabet Σ_A. Then there exists a quasiperiodic SFT $\mathscr{B}$ (over another alphabet Σ_B) of dimension $d+1$, such that $\mathscr{A}$ is* simulated *by $\mathscr{B}$ in the sense of Definition 13.*

(b) Let $\mathscr{A}$ be a $\mathbb{Z}^d$-shift simulated in the sense of Definition 13 by some quasiperiodic SFT $\mathscr{B}$ of dimension $d+1$. Then $\mathscr{A}$ is also quasiperiodic.

Remark 28. Parts (a) and (b) of Theorem 44 are the *if* and *only if* parts of the following characterization: an effective shift is quasiperiodic, if and only if, it is simulated by a quasiperiodic SFT of dimension higher by 1.

Theorem 45. *(a) For every effective minimal $\mathbb{Z}^d$-shift $\mathscr{A}$ there exists a minimal SFT $\mathscr{B}$ in $\mathbb{Z}^{d+1}$ such that $\mathscr{A}$ is simulated by $\mathscr{B}$ in the sense of Definition 13.*

(b) Let $\mathscr{A}$ be a $\mathbb{Z}^d$-shift simulated in the sense of Definition 13 by some minimal SFT $\mathscr{B}$ of dimension $d+1$. Then $\mathscr{A}$ is minimal.

Remark 29. The *only if* part of this theorem (Theorem 45 (b)) is due to the fact that the notion of *simulation* in Definition 13 is rather restrictive. We should keep in mind that in general the d-dimensional subdynamics of a minimal $\mathbb{Z}^{d+1}$-shift is not necessary minimal.

In Section 2.6 we described a very general construction of a self-simulating tile set, and showed that the corresponding SFT enjoys the properties of quasiperiodicity or even minimality. In the previous section we upgraded this construction and superimposed on the generic scheme of self-simulation a new technique: the scheme of embedding in a tiling a sequence from some effective one-dimensional shift. *Á priori*, the new "upgraded" SFT may lose the property of quasiperiodicity. To maintain this property, some additional effort is needed. Such a construction was proposed in [c2]. For more detailed proofs of Theorem 44 and Theorem 45 we refer the reader to [e2].

Theorem 44 implies the following corollary:

Theorem 46. *There exists a quasiperiodic SFT $\mathscr{A}$ of dimension* 2*, such that the Kolmogorov complexity of every $(N \times N)$-pattern in every configuration of $\mathscr{A}$ is $\Omega(N)$.*

Proof. The proof of this theorem is similar to the proof of Theorem 42. We only need to combine Theorem 44 with a fact from [68]: there exists a one-dimensional shift $\mathscr{S}$ that is quasiperiodic, and for every configuration $\mathbf{x} \in \mathscr{S}$ the Kolmogorov complexity of all factors is linear, i.e., $C(x_i x_{i+1} \dots x_{i+n}) = \Omega(n)$ for all i. □

2.11 Robust shifts

2.11.1 Tilings with random errors: Motivation

In this section, we discuss SFT and tilings with local "errors." That is, we accept the configurations where the imposed local rules can be violated. We study "robust" tile sets that enforce some nontrivial structural properties (such as strong aperiodicity or high information density) for all configurations with sparse enough faults.

There are two similar approaches to the definition of a faulty tiling τ. The first approach deals with *errors*: we can consider configurations

$$\mathbf{x} : \mathbb{Z}^2 \to \tau,$$

where for some neighboring cells in $\mathbb{Z}^2$, two neighboring tiles do not match (i.e., two tile from τ have different colors on adjacent sides). The other approach is to allow tilings with *holes*, i.e., configurations where the usual local rules (neighboring tiles must share the same color on the adjacent sides) apply only to neighboring pairs of tiles, without restrictions on the tiles neighboring a *hole*:

Definition 15. *For a subset $E \subset \mathbb{Z}^2$ and a tile set τ we call by a (τ, E)-tiling any mapping*

$$\mathbf{f} : (\mathbb{Z}^2 \setminus E) \to \tau$$

such that for every two neighbor cells $x, y \in \mathbb{Z}^2 \setminus E$, tiles $\mathbf{f}(x)$ and $\mathbf{f}(y)$ satisfy the tiling rules (colors on adjacent sides match). We may say that $\mathbf{f}$ is a τ-tiling of the plane with holes at points of E.

These two approaches are essentially equivalent: we can convert a tiling error into a hole (by deleting one of two non-matching tiles) or convert a one-tile hole into a small group of errors (by substituting an arbitrary tile at the hole). In the approach with holes it is natural to assume that we start with a plane with randomly chosen "gaps" and then try to tile the rest of the plane around the given holes. The approach with errors looks more natural as a simplistic model of crystallization. Since the formal difference between these approaches is very small, we can use both of them as intuitive metaphors.

In all of our constructions of "robust" tilings we show that for some types of tile set, every configuration with sparse enough errors (or holes) can be "repaired" in some sense and converted into a valid tiling without errors. We use the idea of hierarchical hole patching that goes back to Peter Gács, who applied it in a more complicated situation in [24]. The procedure of hierarchical patching means that at first we "repair" small holes that are not too close to each other. This "patching" procedure involves changing a small neighborhood around each of the small holes. This stage of patching typically makes the larger (and not patched yet) holes more isolated, since we eliminated small holes around them. We can then patch some of the larger holes (those that are still not too large and not too close to each other). After that, we repeat the same procedure for even larger holes, etc.

This procedure succeeds (we repair all holes, and only a small fraction of tiles has to be changed during this procedure), assuming that holes are *sparse enough*

in the first place. Below, we specify the technical definition of *sparsity*. This definition is general enough: we can prove that if the set of holes is generated at random (each position becoming a hole independently of other positions with small enough probability ε), then the resulting set is sparse with probability 1.

From the physics viewpoint, this formalism looks rather weak. Indeed, if we think of a physical process of crystallization, then local errors in different positions seem non-independent. However, our approach can be considered as a first simple approximation to the notion of "faulty tilings," until a more adequate formalism is developed.

Holes patching and percolation. The procedure of holes patching in a tiling can be viewed as a generalization of the notion of *percolation* (which is the central concept of percolation theory). Let us consider a very simple tile set that consists of only two tiles: one with all black sides and the other with all white sides. In this example the usual local rules for tilings can be rephrased as follows: each connected component in the complement to the set of holes is either all black or all white.

The procedure of "holes patching" means that we want to make small changes in the tiling that will "patch" the holes and therefore convert the entire plane into black or white. This means that initially, we have either only very few small black island in a white ocean or vice versa, which is a well-known fact from percolation theory: if "errors" are generated at random independently with small probability, then with probability 1 the non-erroneous cells form one giant connected component plus many small islands.

We see from this example that a suitable definition of *sparseness* cannot be too simple. It is not enough to require low density (in the Besicovitch sense) of the set of holes. Indeed, a regular grid of thin lines can have tiny density but still cut the plane into many non-connected squares; if half of these squares are black and another half are white, then we cannot patch the holes with only a small correction.

An appropriate notion of a sparse set can be defined in the framework of algorithmic randomness (Martin-Löf definition of randomness). More specifically, we can define the *sparse* sets as all individual *Martin-Löf random sets* with respect to the Bernoulli distribution B_ε and all subsets of these random sets. This algorithmic notion of "sparseness" was studied in [c21]. It can be proven that every set that is sparse in this sense satisfies the conditions required to implement the iterative procedure of patching the holes. In the present text we do prefer a more direct probabilistic approach.

The formalism of probabilistic quantifiers. Our main results are stated with the help of the probabilistic quantifiers "for almost all" (i.e., with probability 1). The order of quantifiers (existential, universal, and probabilistic) is crucial. Denote with (τ, E)-tiling a tiling of the complement of E, i.e., a configuration

$$\mathbf{f}\colon (\mathbb{Z}^2 \setminus E) \to \tau,$$

where all pairs of neighboring tiles in $\mathbb{Z}^2 \setminus E$ match (i.e., have the same color on the adjacent sides). The statement "a tile set τ is robust" means that *there exists* some $\varepsilon >$

0 such that *for almost all E* (with probability 1 with respect to the distribution where each point belongs independently to E with probability ε) the following property is true:

For every (τ, E)-tiling U *there exists* a τ-tiling U' (of the entire plane) that is "close enough" to U.

In the next section we proceed with several lemmas concerning the structural properties of sparse sets, which are true for almost all randomly chosen sets of errors.

2.11.2 A technical tool for robust tilings (1): hierarchical islands of errors

In this section we develop the notion of "sparsity" of subsets of $\mathbb{Z}^2$. We extensively use the (pretty known) idea of stratification of the set of random errors into isolated "islands" of different size. For a clear explanation of this technique in the context of faulty cellular automata we recommend [50] (a detailed comment on the P. Gács paper [24]). In the subsequent text we follow the exposition in [j7]. A similar argument was used, e.g., in [c22].

Notation: The diameter of a finite set in $\mathbb{Z}^2$ is the maximal distance between its elements. Distance $d(v, w)$ is defined as the maximum of distances along both coordinates of $v, w \in \mathbb{Z}^2$. The r-neighborhood of X is a set of all points y such that $d(y, x) \le r$ for at least one $x \in X$.

The iterative cleaning procedure and islands of errors. Let $E \subset \mathbb{Z}^2$ be a set of points; we say that points in E are *dirty*, and the points in $\mathbb{Z}^2 \setminus E$ are *clean*. Let α and β ($\beta \ge \alpha > 0$) be two integers.

Definition 16. *We say that a non-empty set $X \subset E$ is an* (α, β)-island *in E, if*

- *the diameter of X does not exceed α, and*
- *in the β-neighborhood of X there is no other point from E.*

Let us note that every two islands are disjoint and the distance between their points is greater than β.

Let $(\alpha_1, \beta_1), (\alpha_2, \beta_2), \ldots$ be a sequence of pairs of integers such that $\alpha_i \le \beta_i$ for all i. We define an iterative "cleaning" procedure as follows. At the first step, we find all (α_1, β_1)-islands (*rank 1 islands*) and remove all their elements from E (denote the set of all remaining points by E_1). Then, we find all (α_2, β_2)-islands in E_1 (*rank 2 islands*), remove them, and denote the set of remaining points by $E_2 \subset E_1$, etc. We say that the cleaning process is *successful*, if every dirty point is removed at some stage.

At the ith step, we also keep in mind the β_i-neighborhoods of the islands deleted during this step. We say that a point $x \in \mathbb{Z}^2$ is *affected* during the ith step, if x belongs to one of these neighborhoods.

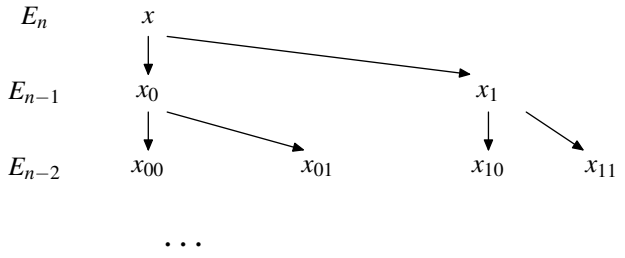

Figure 2.18: Explanation tree; vertical lines connect different names for the same points.

The set E is called *sparse* (for a given sequence of parameters α_i, β_i), if the cleaning process is successful, and, moreover, every point $x \in \mathbb{Z}^2$ is affected at only finitely many steps (in other words, every x is far from islands of sufficiently large ranks).

In what follows we choose the values of α_i and β_i in such a way that for sufficiently small $\varepsilon > 0$, a B_ε-random set is sparse with probability 1. The next lemma provides sufficient conditions for a reasonable choice of parameters.

Lemma 7. *Assume that*

$$\text{for every } n \quad 8\sum_{k<n}\beta_k < \alpha_n \le \beta_n \quad \text{and} \quad \sum_i \frac{\log \beta_i}{2^i} < \infty.$$

Then, for all sufficiently small $\varepsilon > 0$ a B_ε-random set is sparse with probability 1.

Proof. Let us estimate the probability of the event "x is not cleaned after n steps" for a given point x in $\mathbb{Z}^2$. (Obviously, the probability of this event does not depend on x.)

Assume that $x \in E_n$. Then, x belongs to E_{n-1} and is not cleaned during the nth step (when (α_n, β_n)-islands in E_{n-1} are removed; by definition we let $E_0 = E$). Then, $x \in E_{n-1}$ and, moreover, there exists some other point $x_1 \in E_{n-1}$ such that

$$\alpha_n/2 < d(x, x_1) \le \beta_n + \alpha_n/2$$

(note that $\beta_n + \alpha_n/2 < 2\beta_n$). Indeed, if there is no such x_1 in E_{n-1}, then the $(\alpha_n/2)$-neighborhood of x in E_{n-1} is an (α_n, β_n)-island in E_{n-1} and x must be removed at the n-th step of the cleaning procedure.

Further, we apply the same argument on level $(n-1)$. Each of the points x_1 and x (we use notation x_0 for x, to make the notation more uniform) belongs to E_{n-1}; therefore, it belongs to E_{n-2} together with some other point (at a distance greater than $\alpha_{n-1}/2$ but not exceeding $2\beta_{n-1}$). Denote these two other points in E_{n-2} by x_{01} (which exists because $x_0 \in E_{n-1}$) and x_{11} (which exists because $x_1 \in E_{n-1}$) respectively. Thus, we have at least four points denoted by $x_{00} = x_0 = x$, x_{01}, $x_{10} = x_1$, and x_{11} in E_{n-2}. Then, we repeat the same argument for levels $(n-2), (n-3)$, etc. This way we obtain a tree that "explains" why x belongs to E_n, see Fig. 2.18.

The distance between x_0 and x_1 in this tree is at least $\alpha_n/2$, whereas the diameter of the subtrees starting at x_0 and x_1 does not exceed $\sum_{i<n} 2\beta_i$. Therefore, the lemma's assumption guarantees that these subtrees cannot intersect. Since it is true on all levels, all the leaves of the tree are distinct. All 2^n leaves of the tree belong to $E = E_0$. Since each point appears in E independently of other points, every "explanation tree" is valid with probability ε^{2^n}. It remains to estimate the number of possible explanation trees for a given point x.

To specify x_1 we need to specify the difference (in both coordinates) between x_0 and x_1. The distance cannot exceed $2\beta_n$; therefore we need about $2\log(4\beta_n)$ bits to specify the difference between these points (including the sign bits). Then, we need to specify the difference between x_{00} and x_{01} as well as the difference between x_{10} and x_{11}; this requires at most $4\log(4\beta_{n-1})$ bits. To specify the entire tree we therefore need

$$2\log(4\beta_n) + 4\log(4\beta_{n-1}) + 8\log(4\beta_{n-2}) + \cdots + 2^n\log(4\beta_1)$$

bits, which rewrites to

$$2^n(\log(4\beta_1) + \log(4\beta_2)/2 + \cdots).$$

Since the series $\sum \log\beta_n/2^n$ converges by assumption, we can say that the Kolmogorov complexity of an explanatory tree for x is at most $O(2^n)$. Or, in other terms, the total number of explanation trees for a given point (and given n) does not exceed $2^{O(2^n)}$. Hence, the probability for a given point x to be in E_n for a B_ε-random E does not exceed $\varepsilon^{2^n}2^{O(2^n)}$, which tends to 0 (even super-exponentially fast) as $n \to \infty$, assuming that ε is small enough.

We conclude that for a given point x the event "x is not cleaned" has the probability zero. From the countable additivity it follows that with probability 1 all points in $\mathbb{Z}^2$ are cleaned.

It remains to show that every point with probability 1 is affected at only finitely many steps. We note that if x is affected at step n, then some point in its β_n-neighborhood belongs to E_n, and the probability of this event is at most

$$O(\beta_n^2)\varepsilon^{2^n}2^{O(2^n)} = 2^{2\log\beta_n + O(2^n) - \log(1/\varepsilon)2^n};$$

the convergence conditions guarantee that $\log\beta_n = o(2^n)$, so the first term is negligible compared to the others. For small enough ε the probability series converges, and the Borel–Cantelli lemma implies the desired result. □

By definition, a sparse set is split into a union of islands of different ranks. We now prove that these islands collectively occupy only a small part of the plane. To formalize this statement, we use the notion of Besicovitch density.

Definition 17. *Fix a point O of the plane and consider squares of increasing size centered at O. For each square count the fraction of points in this square that belong to E. The* lim sup *of these frequencies is called the* Besicovitch density *of E. (The choice of the center point O does not affect the resulting limit, since for any two points O_1 and O_2, two large squares of the same size centered at O_1 and O_2 share most of their points.)*

By definition, the distance between every two islands rank k is at least β_k. Therefore, the $(\beta_k/2)$-neighborhoods of these islands are disjoint. Each of the islands contains at most α_k^2 points (since an island can be placed in a rectangle with sides α_k). Each neighborhood has at least β_k^2 points (since it contains a $\beta_k \times \beta_k$ square centered at any point of the island). Therefore, the union of all rank k islands has Besicovitch density at most $(\alpha_k/\beta_k)^2$. Indeed, for a large square the islands near its border can be ignored, and all other islands are surrounded disjoint neighborhoods whose density is bounded by $(\alpha_k/\beta_k)^2$, see Fig. 2.19.

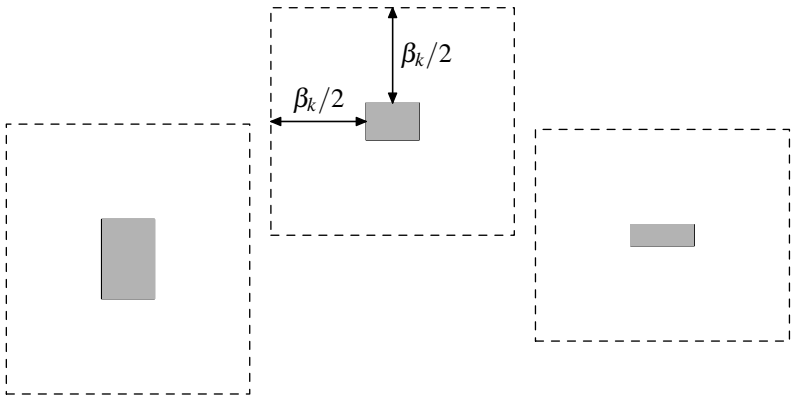

Figure 2.19: Rank k islands form a set of low density. (In this picture each island is shown as a rectangle, which is not always the case.)

We cannot conclude immediately that the overall density of all islands of all ranks does not exceed $\sum_k(\alpha_k/\beta_k)^2$, since the Besicovitch density is not countably subadditive. However, we can prove a slightly weaker bound on the joint density of all islands, if we use the second condition of the definition of a sparse set, which claims that every point of the plane is covered by only finitely many neighborhoods of islands.

Lemma 8. *If E is a sparse set for a given family of α_k and β_k, then the Besicovitch density of E is $O(\sum(\alpha_k/\beta_k)^2)$.*

Proof. Let O be a center point used in the definition of Besicovitch density. By the definition of sparsity, this point is not covered by β_k-neighborhoods of rank k islands for k greater than some K. We now split the set E into two parts: one part ($E_{\le K}$) consists of all islands of a rank at most K and the other part ($E_{>K}$) consists of all islands of bigger ranks.

As seen above, in every large enough square the share of $E_{\le K}$ is bounded by $\sum_{k\le K}(\alpha_k/\beta_k)^2$ up to a minor boundary effect (here we consider each $k \le K$ separately and then sum over all $k \le K$).

A similar bound is valid for every rank k islands with $k > K$, though in this case the argument is different. Indeed, we know that the β_k-neighborhood of every island I does not contain the center point O. Hence, any square S centered at O that

intersects the island, must also contain a significant part of its $(\beta_k/2)$-neighborhood N. More precisely, the intersection of N and S contains at least $(\beta_k/2)^2$ elements, see Fig. 2.20. Therefore, the share of $E_{>K}$ in S is bounded by $4\sum_{k>K}(\alpha_k/\beta_k)^2$. □

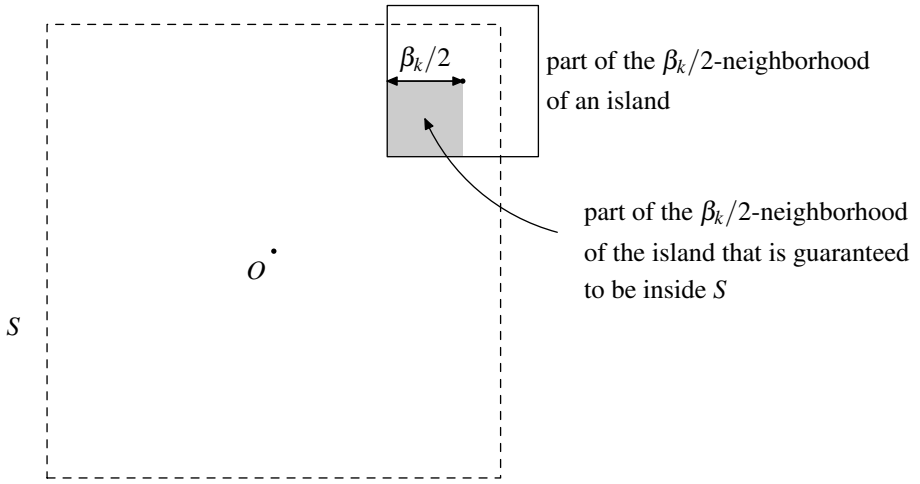

Figure 2.20: Together with a point in a rank k island, every square S contains at least $(\beta_k/2)^2$ points of its $(\beta_k/2)$-neighborhood.

We need a slightly stronger version of Lemma 8. In fact we are interested not only in the Besicovitch density of a sparse set E but also in the Besicovitch density of the union of γ_k-neighborhoods of rank k islands in E. Here γ_k are some parameters; in most applications we set $\gamma_k = c\alpha_k$ for some constant c. The same argument as above gives the bound $4\sum((\alpha_k+2\gamma_k)/\beta_k)^2$. Assuming that $\gamma_k \geq \alpha_k$, we can rewrite this bound as $O(\sum(\gamma_k/\beta_k)^2)$. So we obtain the following statement:

Lemma 9. *If E is a sparse set for a given family of α_k and β_k, and $\gamma_k \geq \alpha_k$, then the union of γ_k-neighborhoods of level k islands (over all k and all islands) has Besicovitch density $O(\sum(\gamma_k/\beta_k)^2)$.*

2.11.3 A technical tool for robust tilings (2): Bi-islands of errors

In Section 2.11.6 we will need a more delicate version of the definition of islands. In fact, we need a counterpart of Lemma 7 that applies even if $\log\beta_n$ grows much faster than 2^n (e.g., for $\beta_n = c^{(2.5)^n}$). To this end we define *bi-islands* (a generalization of the notion of an island from the previous section) and prove for the bi-island some versions of Lemmas 7, 8, and 9.

The revised cleaning procedure and bi-islands of errors. Let $E \subset \mathbb{Z}^2$ be a set of points. As in Section 2.11.2, we say that points in E are *dirty*, and the other points are *clean*. Let $\beta \geq \alpha > 0$ be integers.

Definition 18. *A non-empty set $X \subset E$ is called an* (α, β)-bi-island *in E if X can be represented as the union of some sets X_0, X_1 such that*

- *in the β-neighborhood of $X = X_0 \cup X_1$ there are no points from $E \setminus X$;*
- *the diameters of X_0 and X_1 do not exceed α, and*
- *the distance between X_0 and X_1 does not exceed β,*

see Fig. 2.21.

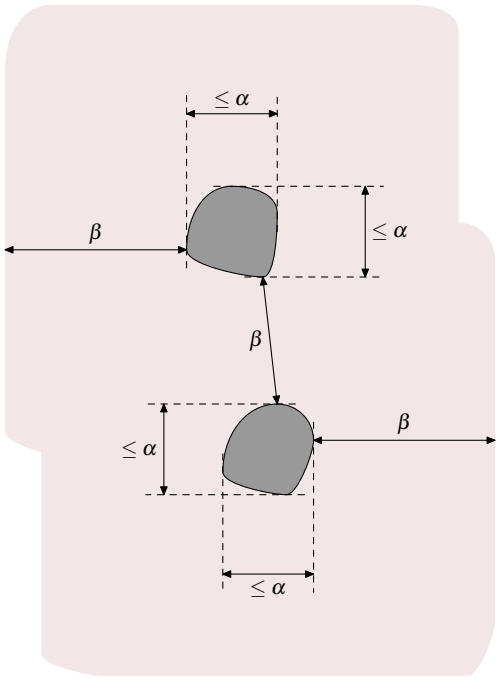

Figure 2.21: A bi-island, a union of two "islands" that are close to each other.

In particular, an (α, β)-island is a special case of an (α, β)-bi-island (we may let X_1 be empty). Note that one and the same bi-island X may be split into X_0 and X_1 in many different ways.

Obviously, every two different bi-islands in E must be disjoint. Moreover, the distance between them must be greater than β. The diameter of every bi-island is at most $(2\alpha + \beta)$.

Let $(\alpha_1, \beta_1), (\alpha_2, \beta_2), \ldots$ be a sequence of pairs of integers, and $\alpha_i \leq \beta_i$ for all i. We define an iterative cleaning procedure for bi-islands. At the first step we find

all (α_1, β_1)-bi-islands and remove all their elements from E; denote the remaining points by $E_1 \subset E$) Then, we find in E_1 all (α_2, β_2)-bi-islands, remove them and denote the remaining points by $E_2 \subset E_1$, etc. We say that the cleaning process is *successful* if every dirty point is removed at some stage.

Similarly to the case of islands, we say that a point $x \in \mathbb{Z}^2$ is *affected* during step i, if x belongs to the β_i-neighborhood of one of the bi-islands of rank i.

The set E is called *bi-sparse* (for given sequences of α_i, β_i), if the cleaning process defined above is successful, and, moreover, every point $x \in \mathbb{Z}^2$ is affected at only finitely many steps (which means that x is far from all bi-islands of sufficiently large ranks).

We choose the values of α_i and β_i in such a way that for sufficiently small $\varepsilon > 0$ a B_ε-random set is bi-sparse with probability 1. The sufficient condition for a successful cleaning procedure becomes weaker than it was in the corresponding statement for islands (Lemma 7):

Lemma 10. *Assume that*

$$\text{for every } n \quad 12 \sum_{k<n} \beta_k < \alpha_n \le \beta_n, \text{ and } \quad \sum_i \frac{\log \beta_i}{3^i} < \infty.$$

Then, for all sufficiently small $\varepsilon > 0$, a B_ε-random set is bi-sparse with probability 1.

Proof. The proof of Lemma 10 is very similar to the proof of Lemma 7. At first, we estimate the probability of the event "x is not cleaned after n steps" for a given point x. If $x \in E_n$, then x belongs to E_{n-1} and is not cleaned during the nth step, when (α_n, β_n)-bi-islands in E_{n-1} are removed. Therefore, $x \in E_{n-1}$. Moreover, we show that there exist *two other* points $x_1, x_2 \in E_{n-1}$ such that the three distances $d(x, x_1)$, $d(x, x_2)$, and $d(x_1, x_2)$ are all greater than $\alpha_n/2$ but not greater than $2\beta_n + 2(\alpha_n/2) < 3\beta_n$.

Indeed, let X_0 be the $(\alpha_n/2)$-neighborhood of x in E. If X_0 were an island, it would be removed. Since this does not occur, there is a point x_1 outside X_0 but in the β_n-neighborhood of X_0.

Let X_1 be the $(\alpha_n/2)$-neighborhood of x_1 in E. Again, the union of X_0 and X_1 is not a bi-island. Both sets X_0 and X_1 have a diameter at most α_n, and the distance between them is at most β_n. So the only reason why they do not form a bi-island is that there exists a point $x_2 \in E$ outside $X_0 \cup X_1$ but in its β_n-neighborhood. The points x_1 and x_2 have the required properties (the distances $d(x, x_1)$, $d(x, x_2)$, and $d(x_1, x_2)$ are greater than $\alpha_n/2$ but not greater than $3\beta_n$).

To make the notation uniform, we denote x by x_0. Each of the points x_0, x_1, x_2 belongs to E_{n-1}. This means that each of them belongs to E_{n-2} together with a pair of other points (at a distance greater than $\alpha_{n-1}/2$ but not exceeding $3\beta_{n-1}$). By continuing this argument we obtain a ternary tree that "explains" why x belongs to E_n.

The distance between every two points among x_0, x_1, and x_2 in this tree is at least $\alpha_n/2$ whereas the diameters of the subtrees, starting at x_0, x_1, and x_2, do not exceed $\sum_{i<n} 3\beta_i$. Thus, the assumption of the lemma guarantees that these subtrees do not intersect and that all the leaves of the tree are different.

The number of leaves in this ternary tree is 3^n, and they all belong to E. Every point appears in E independently of other points; hence, one such "explanation tree" is valid with probability ε^{3^n}. It remains to count the number of all explanation trees for a given point x.

To specify x_1 and x_2 we need to specify the difference in coordinates between x_0 and x_1, x_2. These differences do not exceed $3\beta_n$. Therefore we need about $4\log(6\beta_n)$ bits to specify them. We then need to specify the differences between x_{00} and x_{01}, x_{02} as well as the distances between x_{10} and x_{11}, x_{12} and between x_{20} and x_{21}, x_{22}. This requires at most $12\log(6\beta_{n-1})$ bits. To specify the entire tree we therefore need at most

$$4\log(6\beta_n) + 12\log(6\beta_{n-1}) + 36\log(6\beta_{n-2}) + \cdots + 4\cdot 3^{n-1}\log(6\beta_1)$$

bits, which is equal to

$$4\cdot 3^{n-1}(\log(6\beta_1) + \log(6\beta_2)/3 + \cdots).$$

The series $\sum \log\beta_n/3^n$ converges by the assumption of the lemma. We can conclude that the Kolmogorov complexity of the explanatory tree is $O(3^n)$. Or, more precisely, the total number of explanation trees for a given point (and given n) does not exceed $2^{O(3^n)}$. Thus, the probability for a given point x to be in E_n for a B_ε-random E does not exceed $\varepsilon^{3^n}2^{O(3^n)}$. This values tends to 0 as $n\to\infty$ if ε is small enough.

We conclude that the event "x is not cleaned" (for a given point x) has zero probability; hence, with probability 1 *all* points in $\mathbb{Z}^2$ are cleaned.

It remains to show that every point with probability 1 is affected by only finitely many steps. Indeed, if x is affected at step n, then some point in its β_n-neighborhood belongs to E_n. The probability of this event is at most

$$O(\beta_n^2)\varepsilon^{3^n}2^{O(3^n)} = 2^{2\log\beta_n + O(3^n) - \log(1/\varepsilon)3^n}.$$

From the convergence conditions we have $\log\beta_n = o(3^n)$, so the first term in the exponent is negligible compared to the others. For small enough ε the probability series converges, and the Borel–Cantelli lemma implies our statement. □

We claim that a bi-sparse set occupies only a small part of the plane:

Lemma 11. *Let E be a bi-sparse set for a given family of α_k and β_k. Then, the Besicovitch density of E is $O(\sum(\alpha_k/\beta_k)^2)$.*

Proof. Similar to the proofs of Lemma 8. □

Let γ_k be a sequence of reals. Similar to Lemma 9, we may easily prove that the Besicovitch density of the union of γ_k-neighborhoods of rank k bi-islands (for all k and for all islands) is bounded by $O(\sum(\gamma_k/\beta_k)^2)$. However, in the sequel we need a stronger statement. In Section 2.11.6 we will need a kind of "closure" of the γ_k-neighborhood of a bi-island:

Definition 19. *Let S be a k-level bi-island. We say that $(x,y)\in\mathbb{Z}^2$ belongs to the* extended *γ-neighborhood of S, if there exist two points*

$$(x,y'),(x,y'')\in\mathbb{Z}^2$$

(*with the same first coordinate*) *such that* $\mathrm{dist}(S,(x,y')) \le \gamma$, $\mathrm{dist}(S,(x,y'')) \le \gamma$, *and* $y' \le y \le y''$ *(see Fig. 2.22).*

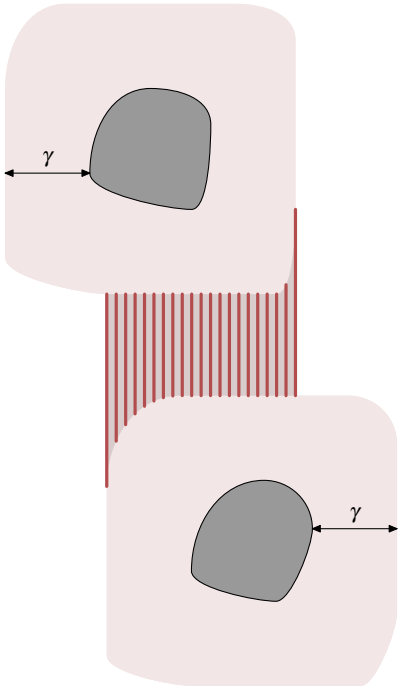

Figure 2.22: An extended neighborhood of a bi-island consists of the neighborhoods of its two parts and a zone between them.

The intuitive meaning of the last definition is quite simple. We take not only the points that are close to a bi-island S but also those points that are placed somehow between the neighborhoods of two parts of S.

Lemma 12. *Let E be a bi-sparse set for a given family of α_k and β_k satisfying the conditions of Lemma 10. Let γ_k be a sequence of numbers, such that $\alpha_k < \gamma_k$, and the series $\sum(\gamma_k/\beta_k)$ converges. Then, the Besicovitch density of the union of extended γ_k-neighborhoods of all rank k bi-islands in E is bounded by $O(\sum(\gamma_k/\beta_k))$.*

Proof. The argument is similar to the proof of Lemma 9. The extended γ_k-neighborhood of a k-level island can be covered by a rectangle of width $O(\gamma_k)$ and height $O(\beta_k+\gamma_k)$. So its area is $O(\gamma_k\beta_k)$ (since $\gamma_k \le \beta_k$). The distance between any two bi-islands of rank k is at least β_k. Hence, the *extended* γ_k-neighborhoods of islands cover the fraction $O(\sum\gamma_k/\beta_k)$ of the plane (*cf.* the bound $O(\sum(\gamma_k/\beta_k)^2)$, which holds for the simple γ_n-neighborhoods). □

The notion a bi-sparse set is used in Section 2.11.6. We apply the definition of a bi-sparse set with parameters α_k, β_k such that $\log \alpha_k \sim q^k$ for $q > 2$, $\beta_k \sim \alpha_{k+1}$, and $\gamma_k = O(\alpha_k)$ or $\gamma_k = O(\alpha_k^2)$. Note that we cannot apply Lemmas 7 and 8 (about *islands*) for these parameters since $\log \beta_k$ grows faster than 2^k.

Remark 30. In the definition of sparse sets in Section 2.11.2, each single island of rank k must be isolated from other islands of rank k. In this section, we extended this definition and introduced the notion of a bi-island. In a similar way, we could define s-islands for any $s \geq 2$ by admitting clusters of s islands of rank k. A set represented as a union of s-islands of different ranks can be called "s-sparse," and a generalization of Lemmas 10 applies to the case of s-islands: a random set is s-sparse with probability 1, if the series $\sum_i \frac{\log \beta_i}{(s+1)^i}$ converges.

We do not develop here the general theory of s-sparse sets, since the concepts of bi-islands and bi-sparsity (i.e., the case $s = 2$) are enough for all of our applications in Section 2.11.6.

2.11.4 A technical tool for robust tilings (3): A self-simulating tiling tolerant to isolated islands of error

In this section we show that there exists an aperiodic tile set where an isolated defect of any size can be healed.

Definition 20. *Let c_1, c_2 be positive integers ($c_1 < c_2$). We say that a tile set τ is* (c_1, c_2)-robust *if for every Δ and for every τ-tiling U of the $(c_2\Delta)$-neighborhood of a square $\Delta \times \Delta$ excluding the square itself there exists a tiling V of the entire $(c_2\Delta)$-neighborhood of the square (including the square itself) that coincides with U outside of the $(c_1\Delta)$-neighborhood of the square, see Fig. 2.23.*

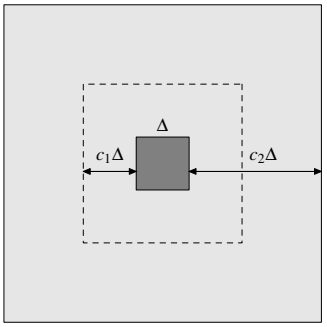

Figure 2.23: Patching a hole of size $\Delta \times \Delta$ given a tiling of the $c_2\Delta$-neighborhood around the hole.

Remark 31. We have already used a similar property of robustness for a part of the tiling representing the space-time diagram of a Turing machine, see p. 110 and p. 121.

Theorem 47. *There exists a self-similar tile set that is* (c_1, c_2)*-robust for some* c_1 *and* c_2.

Proof. For every tile set τ, we can construct a "robustified" version τ' of τ, i.e., a tile set τ' and a mapping $\pi\colon \tau' \to \tau$, such that

(a) π-images of τ'-tilings are exactly τ-tilings and

(b) τ' is "4-robust": Every τ'-tiling of a 4×4 square minus the 2×2 hole in the middle (see Fig. 2.24) can be uniquely extended to the tiling of the entire square of size 4×4.

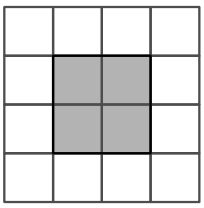

Figure 2.24: A 2×2 hole within a safety border.

To this end we keep in one τ'-tile the information about the 4×4 square in a τ-tiling. Matching rules guarantee that the information contained in every two neighbor τ'-tiles (i.e., the information about a τ-tiling in a rectangle of size 3×4) is consistent. Then, a 2×2 hole is repairable. The tiles at its border (shown in gray in the figure) are consistent and contain all the information that should be held by the missing tiles.

It is not hard to see that this robustification can be combined with the fixed-point construction. That is, we can construct a "4-robust" self-similar tile set τ if the zoom factor N is large enough. It remains to explain that property of 4-robustness implies also (c_1, c_2)-robust for some c_1 and c_2. (The values of c_1 and c_2 depend on N.)

Assume that we are given a tiling of a large enough neighborhood around a $\Delta\times\Delta$ hole. Denote by k the minimal integer such that $N^k \geq \Delta$, and the k-level macro-tiles are greater than the hole under consideration. Note that the size of the k-level macro-tiles is only $O(\Delta)$ since $N^k \leq N\cdot\Delta$.

In the tiling around the hole the structure of $N\times N$ blocks is correct except for the N-neighborhood of the central hole of size $\Delta\times\Delta$ hole. Indeed, the colors of tiles encode coordinates modulo N, so in every connected tiled region the coordinates must be consistent. For a similar reason the structure of blocks of size $N^2\times N^2$ is correct except for the $(N+N^2)$-neighborhood of the hole, etc. Hence, for the value of k chosen above, we have the structure of k-level macro-tiles that is correct except for (at most) $4 = 2\times 2$ squares of level k. We can delete everything in these four squares and use the property of 4-robustness to replace them with macro-tiles that make a valid patch.

To implement this procedure and patch the hole, we need a correct tiling only in the $O(N^k)$ neighborhood of the hole. (More precisely, we need to have a correct tiling in the $(2N^k)$-neighborhood around the hole; as $2N^k \le 2N\Delta$, we set $c_2 := 2N$.) The correction procedure involves changes in another $O(N^k)$-neighborhood of the hole. (More precisely, the changes involve only the (N^k)-neighborhood of the hole; $N^k \le N\Delta$, so we can set $c_1 = N$.) □

2.11.5 Robust self-simulating tilings with variable zoom factors

The construction from the previous section works only for self-similar tilings with a fixed zoom factor. It works fine for simple applications. But in more involved cases we will need self-simulating tilings with variable zoom factors. So now we will explain how to get robust self-simulating tilings with variable zoom factors $N_1, N_2, \ldots$

As well as in the case of a fixed zoom factor, the crucial idea is that the k-level macro-tiles are "responsible" for repairing holes of size comparable to these macro-tiles. Let $\Delta_0 \le \Delta_1 \le \Delta_2 \le \ldots$ be a sequence of integers, and let c_1, c_2 $(c_1 < c_2)$ be positive integers.

Definition 21. *We say that a tile set τ is* (c_1, c_2)-robust against holes of size $\Delta_0, \Delta_1, \ldots$ *if for every n and for every τ-tiling U of the $c_2\Delta_k$ neighborhood around a square of size $\Delta_k \times \Delta_k$ excluding the square itself there exists a tiling V of the entire $c_2\Delta_k$ neighborhood of the square* (*including the square itself*) *that coincides with U outside of the $c_1\Delta_k$ neighborhood of the square.*

The difference from the definition of Section 2.11.4 is that we take into consideration only holes of sizes $\Delta \in \{\Delta_0, \Delta_1, \ldots\}$ instead of holes of arbitrary size.

Lemma 13. *Assume a sequence of zoom factors N_k grows not too fast and not too slow* (*it is enough to assume that $N_k \ge C \log k$ and $C \log N_{k+1} < N_k$ for a large enough C; cf. the discussion in Section 2.5.1*). *Then, there exists a self-simulating tile set with variable zoom factors N_k* (k*-level macro-tiles of size $L_k = N_0 \cdots N_{k-1}$*) *that is* (c_1, c_2)*-robust* (*for some c_1 and c_2*) *against holes of size $L_0, L_1, \ldots$*

Proof. We start with the construction from Section 2.5.1, which gives a self-simulating tile set with variable zoom factors $N_1, N_2, \ldots$ Denote by τ_k the set of k-level macro-tiles corresponding to this tile set.

Further we construct a robustified version of this tile set. To this end we basically repeat the arguments from the proof of Theorem 47. The only difference in the argument is that we now deal with variable zoom factors, and the sizes of holes are taken from the sequence $L_0, L_1, \ldots$

Denote by τ'_k the set of all k-level macro-tiles for the new tiling. We need to find a mapping $\pi\colon \tau'_k \to \tau_k$ such that

(a) π-images of τ'_k-tilings are exactly the τ_k-tilings, and

(b) τ'_k is 4-robust, i.e., every τ'_k-tiling of a 4×4 square minus the 2×2 hole in the middle can be uniquely extended to the tiling of the entire 4×4 square (see again Fig. 2.24) .

To obtain such a robustification, it is enough to keep in every τ'_k-macro-tile the information about a 4×4 square in the τ_k-tiling and use the macro-colors on the borders to guarantee that this information contained in neighboring macro-tiles is coherent.

As usual, this robustification can be combined with the construction of a self-simulating tile set. We get 4-robust macro-tiles for all levels of the construction, and we obtain the required property of generalized robustness. Indeed, assume that a tiling of a large enough neighborhood around a $\Delta \times \Delta$ hole is given, and $\Delta \le L_k$ for some k. In the tiling around the hole, the structure of blocks of size $(L_1 \times L_1)$ is correct everywhere except only for the L_1 neighborhood of the hole. Similarly, the structure of $(L_2 \times L_2)$ blocks is correct except for the $(L_1 + L_2)$ neighborhood of the hole, etc. So we get a k-level structure that is correct except for only $4 = 2 \times 2$ squares of size $L_k \times L_k$. Due to 4-robustness, this hole can be filled with four k-level macro-tiles. Note that we can reconstruct uniquely the ground-level tiles inside a high-level macro-tile when we know he latter's "conscious known" information, i.e., the content of the tape of the Turing machine simulated on the computation zone of this macro-tile. This information can be reconstructed from the consciously known information of the neighboring macro-tiles of the same rank.

To repair the hole, we need only a correct tiling in the $O(L_k)$ neighborhood around this hole. The repairing procedure involves changes in another $O(L_k)$ neighborhood of the hole. More technically, we need to have a correct tiling in the $(2L_k)$-neighborhood of a hole of size L_k, so we can set $c_2 = 2$. Since the correction procedure involves changes in the L_k-neighborhood of the hole, we can set $c_1 = 1$. □

We can make a tiling robust not only against holes, but also against *pairs of holes*. To this end we should slightly modify our definition of robustness. Let

$$\Delta_0 \le \Delta_1 \le \Delta_2 \le \dots$$

be an increasing sequence of integers, and let $c_1 < c_2$ be positive integers.

Definition 22. *We say that a tile set τ is* (c_1, c_2)-robust against pairs of holes of size $\Delta_0, \Delta_1, \dots$, *if the following holds. Let us have two sets $H_1, H_2 \subset \mathbb{Z}^2$, each of which of diameter at most Δ_k (for some $k > 0$). For every τ-tiling U of the $c_2\Delta_k$ neighborhood of the union $(H_1 \cup H_2)$, excluding H_1 and H_2 themselves, there exists a tiling V of the entire $c_2\Delta_k$ neighborhood of $(H_1 \cup H_2)$ (including H_1 and H_2 themselves) that coincides with U outside of the $c_1\Delta_k$ neighborhood of $(H_1 \cup H_2)$.*

Robustification against pairs of holes is done in the same way as the robustification against a single isolated hole. If two given holes are far apart from each other, we can patch them independently; if they are rather close to each other, we patch them as one hole of (roughly) doubled size. So we can use the same robustification technique as before, and we need only to take a large enough "radius of multiplication" D (and use D-robustness instead of 4-robustness). In this way we get the following lemma.

Lemma 14. *Assume a sequence of zoom factors N_k grows not too fast and not too slow (e.g., $N_k \ge C \log k$ and $C \log N_{k+1} < N_k$ for a large enough C). Then, there exists*

a self-simulating tile set with zoom factors N_k (i.e., with k-level macro-tiles of size $L_k = N_0 \cdots N_{k-1}$) that is (c_1, c_2)-robust (for some c_1 and c_2) against pairs of holes of size $L_0, L_1, \ldots$

2.11.6 Robust tilings with high Kolmogorov complexity

In this section we construct a tile set that is robust with respect to random errors and admits only configurations with Kolmogorov complexity $\Omega(n)$ for every pattern of size $n \times n$.

Theorem 48. *There exists a tile set τ and constants $c_1, c_2 > 0$ with the following properties:*

(1) a τ-tiling of $\mathbb{Z}^2$ exists;

(2) in every τ-tiling of the plane every pattern of size $N \times N$ has Kolmogorov complexity at least $c_1 N - C_2$;

(3) for all sufficiently small ε for almost every (with respect to the Bernoulli distribution B_ε) subset $E \subset \mathbb{Z}^2$, every (τ, E)-tiling is at most $1/10$ Besicovitch apart from some τ-tiling of the entire plane $\mathbb{Z}^2$;

(4) for all sufficiently small ε, for almost every B_ε-random subset $E \subset \mathbb{Z}^2$, for every (τ, E)-tiling the Kolmogorov complexity of the centered square of size $N \times N$ in this tiling is $\Omega(N)$.

Preliminary remarks

The main tools of the proof. The rest of the section is devoted to the proof of Theorem 48. It combines the techniques form Section 2.9 and Section 2.11.5. We take a tile set with high complexity tilings from Section 2.9 and then robustify it. The argument is based on the idea of robustness against holes of some sequence of sizes $\Delta_0, \Delta_1, \Delta_2, \ldots$, explained in Section 2.11.5. First of all, we split the set of random errors into bi-islands of different ranks. Then, we eliminate these bi-islands one by one, starting from lower ranks. Each time we correct an isolated bi-island of rank k, we assume that in a large enough neighborhood of this bi-island there are no other errors. Elimination of a k-level bi-island involves corrections in its *extended* $O(\Delta_k)$-neighborhood (with all parameters as specified below).

The main difficulties and ways to circumvent them. We suggest to combine the construction from Section 2.9 with error-correcting methods based on the idea of islands of errors. There are two main difficulties in this plan: fast growing zoom factors and gaps in vertical columns. Let us discuss these two problems in some detail.

The first problem is that our construction of tiling with high Kolmogorov complexity from Section 2.9 requires *variable zoom factors*. What is even worse is that zoom factors N_k must increase very fast (with logarithms growing faster than 2^k). Hence, we cannot directly apply the technique of islands from Section 2.11.2 since

it works only when $\sum \frac{\log \beta_k}{2^k} < \infty$ (β_k must be of the same order as the size of k-level macro-tiles). To overcome this obstacle, we replace the islands with the bi-islands defined in Section 2.11.3.

The second problem is that now we cannot reconstruct a macro-tile from the information "consciously known" to this macro-tile. The missing information is the sequence of bits assigned to the vertical columns (with each vertical column of tiles carrying one bit of an embedded high-complexity sequence). Random errors make gaps in vertical columns, so now the columns are split into disconnected parts, and these parts *a priori* can carry different bits. To overcome this problem, we organize additional information flows between macro-tiles, to guarantee that each infinite vertical column carries in most of its tiles one and the same bit value.

The general scheme of the construction

Let us sketch the general scheme of the construction and fix the values of the main parameters. Our construction combines the techniques from Sections 2.5.1, 2.9, and 2.11.3.

Self-simulating tilings with variable zoom factors (the techniques discussed in Section 2.5.1). We start with the construction of a self-simulating tile set with variable zoom factors $N_k = Q^{\lfloor 2.5^k \rfloor}$ (for a large enough integer $Q > 0$). This means that every k-level macro-tile is an $(N_{k-1} \times N_{k-1})$ array of $(k-1)$-level macro-tiles. The width and the height of a k-level macro-tile measured in tiles are $L_k = N_0 \cdots N_{k-1}$, and $L_k < N_k$. (The constant 2.5 in our construction can be replaced with any rational number between 2 and 3.)

Tilings with high-complexity patterns (techniques from Section 2.9). To guarantee that the valid tilings contain only patterns with high Kolmogorov complexity, we reemploy the construction from Section 2.9 (proof of Theorem 42). Recall that in every valid tiling for this tile set, in the ith column all tiles keep a bit x_i, and the Kolmogorov complexity of every N-bit factor of $\mathbf{x} = (x_i)$ is $\Omega(N)$. This property is guaranteed by computations embedded into the macro-tiles of all levels.

Hierarchical structure of random errors and incremental error correction (techniques from from Section 2.11.3). When we introduce random errors, the construction from Section 2.9 is broken. Indeed, vertical columns can now be split by islands of errors into unconnected parts, and the proof of Theorem 42 does not apply. To make this construction tolerant to errors, we should guarantee that copies of the embedded bits x_i consciously kept by different macro-tiles are coherent, at least for macro-tiles that are not seriously damaged by local errors. To this end, we introduce into the construction checksums, which guarantee that neighboring macro-tiles keep coherent conscious and subconscious information, as we explain in the next section.

To deal with random errors, we use the technique of bi-islands from Section 2.11.3. Our argument works if the diameter of a k-level bi-island is comparable with the size of k-level macro-tiles. More specifically, we let $\alpha_k = 26L_{k-1}$ and $\beta_k = 2L_k$. (The parameters are chosen so that Lemmas 10 and 11 apply.)

The new construction of the tile set

We start with the construction introduced in Section 2.9 (a tile set that guarantees high Kolmogorov complexity for all admissible patterns) and superimpose on k-level macro-tiles some new structures. We define these new structures in four steps.

First step (introducing checksums). For the tile set from Section 2.9 with which we start, every k-level macro-tile M is an $N_{k-1} \times N_{k-1}$ array of $(k-1)$-level macro-tiles; each of these $(k-1)$-level macro-tiles keeps one delegated bit from the embedded sequence. Let us focus on one horizontal row in M in this two-dimensional array of size $N_{k-1} \times N_{k-1}$ (and on the bits assigned to the N_{k-1} macro-tiles of level $k-1$ in this row). Denote the corresponding sequence of bits by $\eta_1, \ldots, \eta_{N_{k-1}}$. We introduce a sort of *erasure code* and calculate some *checksums* for this string of bits. We choose these checksums so that we can reconstruct all bits $\eta_1, \ldots, \eta_{N_{k-1}}$ *if at most D of these bits are erased* (i.e., if we know values η_i for only $N_{k-1} - D$ positions); here, $D > 0$ is a constant (to be fixed later). Besides this combinatorial property of error-correction, we want the checksums to be easily computable. The required checksums can be defined in many ways; we prefer to use one of the most standard and the classic solution from coding theory: the checksums of the Reed–Solomon code.

Let us go into more detail. For every k we fix a finite field $\mathbb{F}_k$ with at least $N_{k-1} + D$ elements. Then, we calculate a polynomial of degree less than N_{k-1} that takes values $\eta_1, \ldots, \eta_{N_{k-1}}$ at some specific N_{k-1} points of the field. Further, we take as checksums the values of this polynomial at some other D points from the field. (These $(N_{k-1} + D)$ points of the field must be fixed in advance). Two different polynomials of degree less than N_{k-1} can coincide in at most $(N_{k-1} - 1)$ points. Hence, if D bits from the sequence $\eta_1, \ldots, \eta_{N_{k-1}}$ are erased, we can reconstruct them, given the remaining (non-erased) bits η_j and the checksums defined above.

The defined checksums contain $O(\log N_{k-1})$ bits of information. In what follows we discuss how to compute them.

Second step (calculating checksums). We can compute the checksums while going from left to right along the sequence $\eta_1, \ldots, \eta_{N_{k-1}}$ in a rather standard way as follows. Indeed, let $\eta_1, \ldots, \eta_{N_{k-1}}$ be the values of a polynomial $p(x)$ (of degree less than N_{k-1}) at points $x_1, \ldots, x_{N_{k-1}}$. Assume we want to reconstruct all coefficients of this polynomial. We can do this by the following iterative procedure. For $i = 1, \ldots, N_{k-1}$ we calculate polynomials (as a list of coefficients) $p_i(x)$ and $q_i(x)$ (of degree $\le (i-1)$ and i, respectively) such that

$$p_i(x_j) = \eta_j \text{ for } j = 1, \ldots, i$$

and

$$q_i(x) = (x - x_1) \cdots (x - x_i).$$

It is not hard to see that for each i, polynomials p_{i+1} and q_{i+1} can be derived from polynomials p_i and q_i, and the values x_{i+1} and η_{i+1}.

In fact, computing the entire polynomials is wasteful, since we need as the final result only the checksums (which are values of $p = p_{N_{k-1}}(x)$ at D points of the field).

If we only want to get the value $p(a)$ at some particular point a, then we can perform the same calculations modulo $(x-a)$, which is much cheaper. To obtain the value of $p(x)$ at D different points, we run in parallel D copies of this process. At each step of the computation we need to keep in memory only $O(1)$ elements of $\mathbb{F}_k$, i.e., only $O(\log N_{k-1})$ bits of temporary data (the multiplicative constant in these $O(\cdot)$ notations depends on D).

We now embed the above procedure into the computation zones of $(k-1)$-level macro-tiles. In each row of length N_{k-1} in a k-level macro-tile, the partial results of the calculation are transferred from one $(k-1)$-level macro-tile to another one, from the left to the right. For each row, the final result of this computation is embedded in the conscious information (bits on the tape of the Turing machine in the computation zone) of the rightmost $(k-1)$-level macro-tile of the row.

To make this work, we include into the conscious memory of $(k-1)$-level macro-tiles additional $O(\log N_{k-1})$ bits of information and add the same number of bits to their macro-colors. This fits easily the construction of a self-simulating tiling, since zoom factors N_k grow fast and we have enough room in the computation zone.

Third step (consistency of checksums between macro-tiles). So far, we have modified our tile set so that each row in a k-level macro-tile contains $O(\log N_{k-1})$ bits of checksums. We now want to guarantee that these checksums are the same for every two vertically neighboring macro-tiles. This property is automatically true for a valid tiling of the plane: the checksums are computed from the delegated bits (which come from the sequence of bits encoded into tiles of the ground level), and so the corresponding checksums for all vertically aligned macro-tiles are equal to each other. Nonetheless, this property can fail for a tiling with errors. To make it error-resistant, we must revise again the data embedded in macro-tiles.

It is inconvenient to keep the checksums for each row only in the rightmost cell of this row. We spread this information across many cells, by propagating the checksums of the ith row in a k-level macro-tile M ($i = 1,\dots,N_{k-1}$) along the entire ith row and along the entire ith column of M. In other words, these checksums must be "consciously" known to all $(k-1)$-level macro-tiles in the ith row and in the ith column of M. In Fig. 2.25 we show the area of propagation of checksums for two rows (the ith and the jth rows).

On the border between two neighboring k-level macro-tiles (one above another) we check that in each column $i = 1,\dots,N_{k-1}$ the checksums computed in both macro-tiles match. These sanity checks are redundant if there are no errors in the tiling; but they are useful to resist errors, as we explain below.

Fourth step (robustification). The explained above features arranged in every k-level macro-tile (bits delegating, computing and propagating checksums, and all other computations simulated in the computation zone of a macro-tile) are simulated by means of bits kept in the "consciousness" (i.e., in the computation zone) of $(k-1)$-level macro-tiles. We now fix some constant C and "robustify" this construction in the following sense. We require that each $(k-1)$-level macro-tile M keeps in its consciousness not only "its own" data, but also the bits previously as-

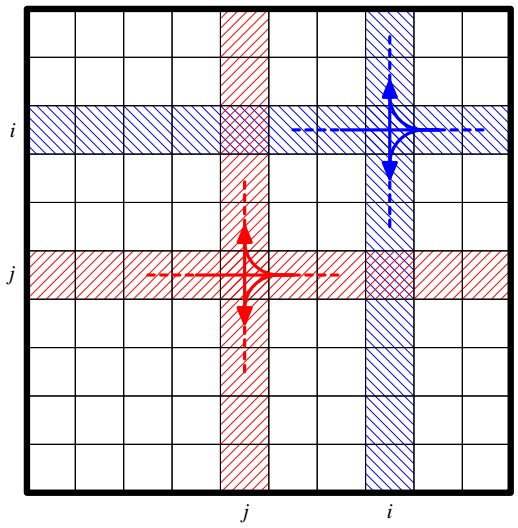

Figure 2.25: Propagation of checksums inside a macro-tile.

signed to all $(k-1)$-level macro-tiles from its $(C \cdot L_{k-1})$-neighborhood (i.e., to the $(2C+1) \times (2C+1)$ array of $(k-1)$-level macro-tiles centered at M). Note that this revision requires multiplying the size of the conscious memory of each macro-tile by only a constant factor. We embed the new conscious memory of macro-tiles in their macro-colors: neighboring macro-tiles check that the data in their consciousness are coherent.

We choose the constant C so that every k-level bi-island (which consist of two parts of size α_k) and even the $\gamma_k = O(\alpha_k)$-neighborhood of every k-level bi-island (we specify γ_k below) can involve only a small part of the (CL_{k-1})-neighborhood of any $(k-1)$-level macro-tile. (Here we talk about neighborhoods, not about extended neighborhoods of bi-islands.)

For the revised tile set, we can reconstruct the conscious memory of a k-level macro-tile and of its $(k-1)$-level sons when this macro-tile is damaged by one k-level bi-island (assuming there are no other errors).

The last remark (the number of bits in the consciousness of a macro-tile). In the revised construction we put into the computation zones of every $(k-1)$-level macro-tile only poly$(\log N_{k-1})$ bits of data. We note again that this requirement fits well our construction of a self-simulating tiling, since poly$(\log N_{k-1})$ is much less than N_{k-2}, and we have enough room to keep and process all these data.

The tile set τ is now defined. Clearly, τ-tilings exist. Since every τ-tiling involves a sequence $\mathbf{x} = (x_0)$ with Levin's property, it follows that every $N \times N$ pattern of every τ-tiling has Kolmogorov complexity $\Omega(N)$. In what follows we prove that this τ also satisfies Claim (3) of Theorem 48.

Error-correcting procedure

Let τ be the tile set described in Section 2.11.6, and let $\varepsilon > 0$ be a small enough real. Lemma 2.21 implies that a B_ε-random set with probability 1 is bi-sparse. We fix a bi-sparse set $E \subset \mathbb{Z}^2$ (for the values of α_i and β_i chosen above), and a τ-tiling T of $\mathbb{Z}^2 \setminus E$. In this section we explain how to convert T into a tiling T' of the entire plane so that T' remains close enough to T.

Since E is bi-sparse, it can be represented as a union of isolated bi-islands of different ranks. We now correct these bi-islands one by one, starting from bi-islands of low ranks. To prove that this process converges, we need to understand the basic step of this procedure, i.e., the correction of one bi-island S of rank k (assuming that the β_k-neighborhood of this bi-island is free of other errors).

By definition every k-level bi-island S is a union of two clusters S_0, S_1, and the diameters of both S_0 and S_1 are at most $\alpha_k = O(L_{k-1})$. These clusters touch only $O(1)$ macro-tiles of level $(k-1)$. The distance between S_0 and S_1 is at most β_k, and we assume that the β_k-neighborhood of S has already been cleaned of other errors. In what follows we perform a correction procedure around S that involves only points in the extended γ_k-neighborhood of S, where $\gamma_k = 2\alpha_k$.

Let M be a k-level macro-tile intersecting the extended γ_k-neighborhood of S. We want to find a repaired version of M. Basically, we need to reconstruct the "correct" versions of all $(k-1)$-level macro-tiles inside M damaged by S. We start by reconstructing the conscious information in all $(k-1)$-level macro-tiles in M. This is enough to get all bits of the embedded sequence $\mathbf{x}$ from the "zone of responsibility" of M. Next, we consistently reconstruct all n-level macro-tiles inside M for all $n < k$.

The consciousness memory (the data in the computation zone) of every $(k-1)$-level macro-tile M' consists of several data fields (see Section 2.8):

[A] the binary representation of the number $(k-1)$ and of the coordinates of M' in the father macro-tile M;

[B] the bits used to simulate a Turing machine on the computation zone of M and the bits used to implement the communication wires of M;

[C] the bit (from the embedded sequence $\mathbf{x}$) delegated to M';

[D] the bit (from $\mathbf{x}$) delegated to M;

[E] the bits used to calculate and communicate the checksums for the corresponding row of $(k-1)$-level macro-tiles in M; and

[F] a factor from $\mathbf{x}$ to be checked; M' verifies on its computation zone that this factor does not contain any subword of low Kolmogorov complexity.

The corrected version of [A] can be reconstructed from the surrounding macro-tiles of the same level. Fields [B], [C], [D], and [E] can be reconstructed, due to robustification on the level of $(k-1)$-level macro-tiles. (We can consistently recover these fields for any $C \times C$ group of missing or corrupt $(k-1)$-level macro-tiles.)

A trickier task is recovering of [F]. To this end, we need to reconstruct the bits of the embedded sequence $\mathbf{x}$ from the zone of responsibility of M. We recover these bits given the neighbor k-level macro-tiles above or below M. (We known that S touches only $O(1)$ k-level macro-tiles, and around them there is a clean area of k-level macro-tiles that are free of errors.) However, we must prove that the bits of the embedded sequence remain consistent above M, below M, and inside M. This is the stage where we use the checksums.

Denote by M_u and M_d (the "up" and "down" neighbors) the k-level macro-tiles just above and below S, see Fig. 2.26. Both M_u and M_d must be free of errors, since the distance between S and other k-level bi-islands is greater than $\beta_k = 2L_k$. In what follows we discuss the case shown in to Fig. 2.26, where bi-island S touches only one k-level macro-tile. (If S touches several k-level macro-tiles, substantially the same arguments work.) It is enough to prove that the embedded bits of $\mathbf{x}$ assigned to the columns of M_u and in M_d match each other (for each i, the i-th column of M_u carries the same embedded bit as the i-th column of M_d).

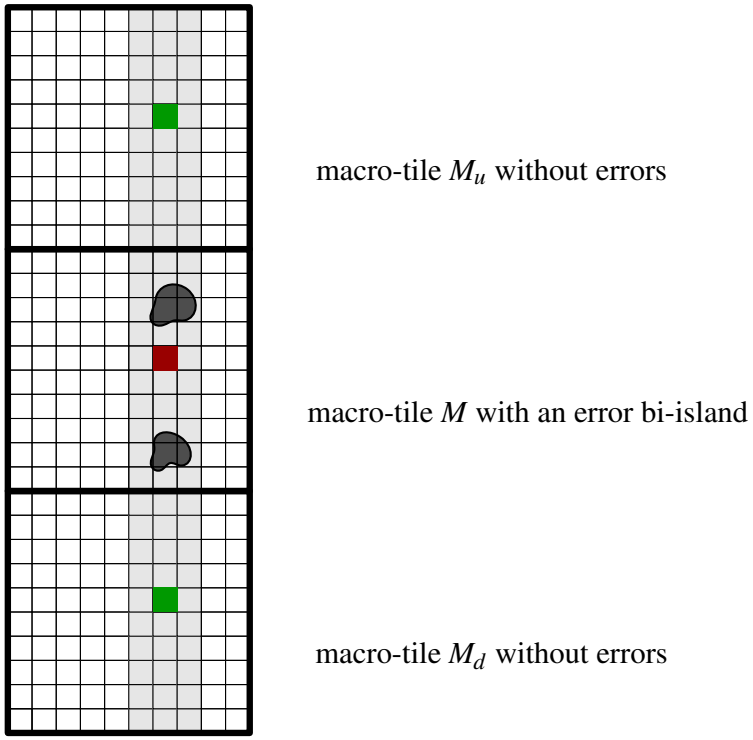

Figure 2.26: Bi-island of errors in a macro-tile.

The macro-tiles M_u and M_d are error-free; therefore, the sequences of L_k bits x_i assigned to the columns of these macro-tiles are well defined. Since there are no errors, the conscious information (including checksums) in all macro-tiles of all levels

inside M_u and M_d is consistent with these bits. Hence, these bits x_i are correctly delegated to the corresponding $(k-1)$-level macro-tiles inside M_u and M_d. However, it is not evident that the sequences of L_k bits embedded in M_u and in M_d are equal to each other.

We start with an observation that bit sequences for M_u and M_d coincide with each other at most positions. Indeed, they match for all columns (from the range $0,\dots,L_k-1$) that do not intersect bi-island S (if a column of tiles is not damaged on the ground level, then the assigned bit spreads correctly though macro-tiles M_u, M, and M_d). Thus, the bits delegated to the corresponding $(k-1)$-level macro-tiles inside M_u and M_d are equal to each other except for only $(k-1)$-level macro-tiles in the "gray zone" in Fig. 2.26, which contains the vertical stripe of columns that go through the $(k-1)$-level macro-tiles involved in the correction of S. The width of this stripe is $O(1)$ macro-tiles of level $(k-1)$. Hence, for $i=0,\dots,(N_{k-1}-1)$, if we compare the ith rows of $(k-1)$-level macro-tiles in M_u and in M_d, we see that the sequences of delegated bits are equal to each other except possibly for only $O(1)$ bits (delegated to $(k-1)$-level macro-tiles in the gray zone).

Due robustness, the checksums are correctly transmitted through M. Hence, for every i, the checksums computed for the i-th rows in M_u and in M_d must be equal to each other. Thus, for the i-th rows of $(k-1)$-level macro-tiles in M_u and in M_d we know that (a) all except $O(1)$ delegated bits match, and (b) the checksums match. From the property of the chosen erasure code it follows that in fact all delegated bits in these rows match without any exceptions. Therefore, the bits of $\mathbf{x}$ embedded in M_u and in M_d (on the ground level) are the same. So we can use these bits to repair M and obtain a consistent tiling around it.

We are almost done. Bi-island S is corrected; we reconstructed conscious information for the k-level macro-tile M and for all its $(k-1)$-level sons. We now can reconstruct fields [F] in the damaged $(k-1)$-level macro-tiles inside M. To this end, we just take the corresponding bits x_i from the zone of responsibility (shared by M, M_u, and M_d). It remains to explain why the checking procedure does not fail for these groups of bits (i.e., $(k-1)$-level macro-tiles do not discover in these bit strings any factors of low Kolmogorov complexity). But this is true because the macro-tiles of levels $(k-1)$ (and below) inside M apply exactly the very same checks to the very same factors of $\mathbf{x}$ as their homologue macro-tiles in M_u and M_d. Since there are no errors in M_u and M_d, these checks are all successful.

Let us inspect again the correction procedure explained above and observe which tiles are involved in the error-correcting process around S. We do not change the $(k-1)$-level macro-tiles outside the gray zone in Fig. 2.26. In the gray zone only the part between two clusters of S (and their small neighborhoods) is affected. Indeed, for the tiles of M above S, the assigned bits x_i are the same as in the corresponding columns of M_u; in the tiles of M below S the assigned bits x_i are the same as in the corresponding columns of M_d. Hence, we do not need to modify the subconscious information of $(k-1)$-level macro-tiles that are above or below S. Thus, the area involved in the correction procedure is covered by the extended neighborhood of S. (In fact, this argument is the motivation of our definition of extended neighborhood.)

We conclude that the incremental correction procedure eliminates all bi-islands

of errors and only the extended γ_k-neighborhoods of k-level bi-islands are involved in this process. Thus, Claim (3) of Theorem 48 follows from Lemma 12.

Lower bound of complexity for a tiling with random errors

It remains to prove Claim (4) of Theorem 48. We have shown above that if the set of errors E is bi-sparse, then every (τ, E)-tiling T can be converted into a τ-tiling T' of the entire plane, and the difference between T and T' is covered by the extended γ_k neighbors of k-level bi-islands from E ($k = 0, 1, \ldots$). We want to prove now that in the initial tiling T the Kolmogorov complexity of the centered squares of size $N \times N$ was also $\Omega(N)$.

Fix a point O in the plane. Since E is bi-sparse, O is covered by β_k-neighborhoods of only finitely many bi-islands. We conclude that for all large enough Δ, if the $\Delta \times \Delta$ squares Q_Δ centered at O intersect the extended γ_k-neighborhoods of a k-level bi-island, then $\beta_k < \Delta$. (If the extended γ_k-neighborhood of a bi-island intersects Q_Δ and $\beta_k \geq \Delta$, then $\beta_k - \gamma_k > \Delta/2$, and O is covered by the β_k-neighborhood of this bi-island.) Therefore, to reconstruct T' in Q_Δ from the original tiling from T, it is enough to correct in T the bi-islands of level k such that $\beta_k < \Delta$.

To reconstruct T' in Q_Δ we need to know the original tiling T in Q_Δ and in some neighborhood around Q_Δ (i.e., in a centered $O(\Delta) \times O(\Delta)$ square $Q_{\Delta'}$, which is greater than Q_Δ by only a constant factor). Indeed, given the tiling T restricted on $Q_{\Delta'}$, we can locally correct in this part of the tiling all bi-islands of levels $1, 2, \ldots, k$ (such that $\beta_k < \Delta$) one by one. The local correction of a bi-island of errors restricted on $Q_{\Delta'}$ gives the same results as we would obtain in the error-correcting procedure on the entire plane $\mathbb{Z}^2$, unless this bi-island is too close to the border of $Q_{\Delta'}$ (in this exceptional case the local correction procedure should involve the information outside $Q_{\Delta'}$). Thus, given tiling T inside $Q_{\Delta'}$ we can reconstruct T' not in the entire $Q_{\Delta'}$ but at all points that are far enough away from the border of this square. If $\Delta' = c\Delta$ for a large enough c, then the original tiling in $Q_{\Delta'}$ provides enough information to reconstruct T' in Q_Δ.

We know that the Kolmogorov complexity of error-free tiling T' in Q_Δ is $\Omega(\Delta)$. Therefore, the Kolmogorov complexity of the original T-tiling in the greater square $Q_{\Delta'}$ is also $\Omega(\Delta)$. Since Δ' is greater than Δ by only a constant factor, we conclude that the Kolmogorov complexity of the (τ, E)-tiling T, restricted to the centered $(\Delta' \times \Delta')$ square, is $\Omega(\Delta')$. This concludes the proof of Theorem 48.

2.11.7 Robust strongly aperiodic tilings

We now can construct a robust strongly aperiodic tile set (a robustified version of Theorem 43).

Theorem 49. *There exist a constant $\alpha > 0$ and a tile set τ with the following properties:*

(1) τ-tilings of $\mathbb{Z}^2$ exist and

(2) *for all sufficiently small* ε *for almost every* (*with respect to* B_ε) *subset* $E \subset \mathbb{Z}^2$ *every* (τ, E)*-tiling is at least* $1/10$ *Besicovitch apart from every periodic mapping* $F : \mathbb{Z}^2 \to \tau$.

Remark 32. Since the tiling contains holes, we need to specify how we treat these holes when defining the Besicovitch distance. We do *not* count points in E as points where two mappings differ; this makes our statement stronger.

Proof. Similar to the proof of Theorem 43, we deduce strong aperiodicity from high Kolmogorov complexity. Indeed, due to Theorem 48, we have tile set ρ that is robust with respect to random errors; we known also that for every ρ-tiling every horizontal row of tiles embeds a sequence of bits (one and the same for each row) whose factors have high Kolmogorov complexity. Such a sequence cannot match itself well after a non-zero translation. Technically, we can claim that 49% of bits must be changed by a non-zero translation.

We now combine this tile set ρ with a similar 90°-rotated construction. For the new tile set τ, any non-zero translation shifts either a vertical or a horizontal sequence embedded in a tiling, and therefore it changes a constant fraction (at least 49%) of the positions.

For almost all B_ε-random sets $E \subset \mathbb{Z}^2$, every (ρ, E)-tiling is at most $1/10$ Besicovitch apart from some ρ-tiling of the entire plane $\mathbb{Z}^2$. The same property is true for the version of ρ rotated by 90°. Hence, for the obtained tile set τ a (ρ, E)-tiling is at most $1/5$ Besicovitch apart from some τ-tiling of the entire plane. As we observed above, in a tiling of the plane every translation changes at least 49% of tiles. Hence, every translation changes changes a constant fraction of tiles in a (ρ, E)-tiling (before repairing the errors). □

Remark 33. The claim of the theorem could be made stronger: its claim is true for any constant $\alpha < 1$. To this end, we should embed in a tiling a sequence with Levin's property over a large enough alphabet.

Perspectives and Future Work

In this section we briefly discuss possible directions for the future work. We mention three types of problems: information inequalities and their applications, combinatorial properties of random and pseudo-random objects, and algorithmic techniques in questions concerning multidimensional shifts.

1. Information inequalities We propose to proceed with the research of non-classical information inequalities, focusing on information inequalities with applications in network coding, combinatorics, and communication complexity.

1.1. The geometry of the cones of the universal linear inequalities for Shannon's entropies of $n \geq 4$ random variables is still not well understood. It is known that these cones are non-polyhedral (F. Matúš), however we still have no satisfactory understanding of their structure and the geometric properties of their surfaces. From this perspective, the further study of conditional information inequalities seems to be promising.

1.2. As of now, there are two general techniques of deducing non-Shannon-type information inequalities: the method of vanishing the conditional mutual information (which appeared in works by Z. Zhang and R.W. Yeung, and was later developed by L. Csirmaz, F. Matúš, and others) and the method of common information based on the Ahlswede–Körner lemma (this technique was suggested in our work with K. Makarychev, Yu. Makarychev, and N. Vereshchagin). Both techniques can be pretty well formalized, they both lead to efficient computer-aided procedures helping to search for new information inequalities.

Until now, the known applications of non-Shannon-type information inequalities followed a two step scheme: firstly, an extensive search of new inequalities (using one of the techniques mentioned above), and then another search over all found inequalities and their combinations to achieve the best result in the desired application. This approach usually involves extremely heavy computations, and in most applications in question the limits of the modern computers are about to be reached. However, we could use another strategy of computer-aided proofs with information inequalities: it seems to be possible to merge both stages of the brute-force search and adjust the procedure of inference of new inequalities, so that only inequalities helpful for this specific application are involved. This approach requires a better understanding of the "physical meaning" of new inequalities and more sophisticated *ad hoc* search algorithms for every specific problem. We believe that

these techniques would extend the area of applications of information inequalities. The promising targets for this approach are problems of secret sharing and network coding.

1.3. We propose to study more elaborately the information inequalities for resource-bounded versions of Kolmogorov complexity. There is no hope for progress with programs that run in polynomial time; L. Longpré and S. Mocas showed that under plausible assumptions from computational complexity even the property of symmetry of the mutual information fails for poly-time bounded Kolmogorov complexity. At the same time, natural counterparts of all basic information inequalities hold true for exp-time and poly-space bounded variants of Kolmogorov complexity. The question of non-Shannon-type inequalities for these complexity measures remain widely open. Filling this lacuna and exploring possible applications of information inequalities in computational complexity is an interesting direction of research.

1.4. It seems to be interesting to proceed with the research of combinatorial applications of constraint information inequalities. Recent results suggest that a suitable domain of these applications is graph theory and, more specifically, clique covers and biclique covers for different classes of graphs. The purpose of this research is to better understand the relation between the information-based arguments and more conventional bounds for the sizes of clique covers. We also expect that the clique cover bounds based on information inequalities can be applied in communication complexity (for randomized and non-deterministic communication protocols).

2. Pseudo-random objects The study of random graphs is a large and well developed branch of graph theory. It deals with the "typical" properties that hold with high probability for a randomly chosen graph. For example, it is known that the vast majority of graphs of fixed degree are *expanders* (the graphs with properties of high expansion, strong connectivity, fast mixing, and so on). Graphs of this type have many important applications in computer science and coding theory. The difficulty is that, though almost any randomly chosen graph is an expander, it is rather hard to produce such a graph explicitly and deterministically.

We propose to study an intermediate approach that combines the usual deterministic and probabilistic paradigms; we suggest to produce graphs using (more or less conventional) pseudo-random number generators. The idea is to compare the properties (expansion rate, spectral gap) of pseudo-random graphs with the typical properties of truly random graphs. Both positive and negative results would be interesting; either we obtain efficient constructions that produces pseudo-random graphs with nice combinatorial properties (useful for applications), or we discover new tests to distinguish truly random sequences from pseudo-random ones.

3. Multidimensional shifts Here, we outline possible directions of future work concerning multidimensional shifts (the questions of information density and the problem of robustness).

3.1. We suggest to proceed with the research of different versions of the information density in shifts of finite type and related questions of symbolic dynamics. In

particular, we believe that time-bounded Kolmogorov complexity is a suitable measure of information density to discern the elusive edge between sofic and non-sofic effective shifts.

3.2. It seems to be interesting to study mathematical models of origins and stability of non-periodic structures. The model of "faulty tilings" explored in our joint work with B. Durand and A. Shen is apparently not suitable to describe physical phenomena. We suggest to study the models of "robust" and "self-correcting" tilings that are more relevant to mathematical physics (e.g., the models based on Gibbs measures). This area remains pretty much uncharted waters, and the implied questions are probably very difficult. Nevertheless, we suggest to try to apply our algorithmic techniques to this domain.

Bibliographie Personnelle[3]

Revues internationales à comité de lecture

[j1] D. Chumbalov, A. Romashchenko. *On the Combinatorial Version of the Slepian-Wolf Problem.* IEEE Transactions on Information Theory, vol. 64, issue 9 (2018) pp. 6054-6069.

[**j2**] T. Kaced, A. Romashchenko, N. Vereshchagin. *Conditional Information Inequalities and Combinatorial Application.* IEEE Transactions on Information Theory, vol. 64, issue 5 (2018) pp. 3610-3615.

[j3] A. Romashchenko and A. Shen. *Topological Arguments for Kolmogorov Complexity.* Theory of Computing Systems, vol. 56, issue 3 (2015) pp. 513–526.

[j4] L. Bienvenu, A. Romashchenko, A. Shen, A. Taveneaux, S. Vermeeren. *The axiomatic power of Kolmogorov complexity.* Annals of Pure and Applied Logic, vol. 165, issue 9 (2014), pp. 1380–1402.

[**j5**] A. Romashchenko. *Pseudo-random graphs and bit probe schemes with one-sided error.* Theory of Computing Systems, vol. 55, issue 2, (2014) pp. 313–329.

[**j6**] T. Kaced, A. Romashchenko. *Conditional Information Inequalities for Entropic and Almost Entropic Points.* IEEE Transactions on Information Theory, vol. 59, issue 11, (2013) pp. 7149–7167.

[**j7**] B. Durand, A. Romashchenko, A. Shen. *Fixed-point tile sets and their applications.* Journal of Computer and System Sciences. vol. 78, issue 3 (2012) pp. 731–764.

[**j8**] D. Musatov, A. Romashchenko, A. Shen. *Variations on Muchnik's Conditional Complexity Theorem.* Theory of Computing Systems. vol. 49, issue 2 (2011) pp. 227–245. (version préliminaire dans *Proc. of CSR 2009*, voyez ci-dessous)

[**j9**] An. Muchnik, A. Romashchenko. *Stability of properties of Kolmogorov complexity under relativization.* Problems of Information Transmission, vol. 46, issue 1 (2010) pp. 38–61.

[**j10**] B. Durand, A. Romashchenko, A. Shen, *Fixed point theorem and aperiodic tilings.* The Logic in Computer Science Column by Yuri Gurevich. Bulletin of the EATCS, vol. 97 (2009) pp. 126–136.

[j11] T. Lee, A. Romashchenko. *Resource Bounded Symmetry of Information Revisited.* Theoretical Computer Science, vol. 345, issue 2–3 (2005), pp. 386–405.

[**j12**] A. Romashchenko. *A Criterion of Extractability of Mutual Information for a Triple of strings.* Problems of Information Transmission, vol. 39 (2003), issue 1, pp. 148–157.

[**j13**] K. Makarychev, Yu. Makarychev, A. Romashchenko, N. Vereshchagin. *A New Class of non Shannon type inequalities for Entropies.* Communications in Information and Systems (2002), issue 2, pp. 147–166.

[3] Les références mentionnées dans le manuscrit sont indiquées en gras.

[**j14**] A. Romashchenko, A. Shen, N. Vereshchagin. *Combinatorial Interpretation of Kolmogorov Complexity.* Theoretical Computer Science, vol. 271 (2002), pp. 111–123. (version préliminaire dans *Proc. of 15th Annual IEEE Conference on Computational Complexity*, voyez ci-dessous)

[**j15**] A. Chernov, An. A. Muchnik, A. Shen, A. Romashchenko, N. Vereshchagin. *Upper semi-lattice of binary strings with the relation x is simple conditional to y.* Theoretical Computer Science, vol. 271 (2002), pp. 69–95. (version préliminaire dans *Proc. of 14th Annual IEEE Conference on Computational Complexity*, voyez ci-dessous)

[**j16**] D. Hammer, A. Romashchenko, A. Shen, N. Vereshchagin. *Inequalities for Shannon Entropy and Kolmogorov Complexity.* Journal of Computer and System Sciences, vol. 60 (2000), pp. 442–464. (version préliminaire dans *Proc. of 13th Annual IEEE Conference on Computational Complexity*, voyez ci-dessous)

[**j17**] A. Romashchenko. *Pairs of Words with Nonmaterializable Mutual Information.* Problems of Information Transmission, vol. 36 (2000), issue 1, pp. 1–18.

[j18] A. Romashchenko. *Sequences of binary strings with relation of conditional simplicity.* Moscow University Mathematics Bulletin, 55 (2000), issue 5, pp. 20–22.

Actes de conférences les plus significatives avec comité de lecture

[**c1**] A. Romashchenko and M. Zimand. *An operational characterization of mutual information in algorithmic information theory.* Proceedings of the 45th International Colloquium on Automata, Languages, and Programming (ICALP), 2018. Track A. 95:1-95:14. (version étendue dans *Electronic Colloquium on Computational Complexity*, voyez ci-dessous)

[**c2**] B. Durand, A. Romashchenko. *On the expressive power of quasiperiodic SFT.* Proc. 43 International Symposium on Mathematical Foundations of Computer Science (MFCS) (2017), 5: pp. 1–14.

[c3] D. Itsykson, A. Knop, A. Romashchenko, D. Sokolov. *On OBDD-Based Algorithms and Proof Systems That Dynamically Change Order of Variables.* Proc. 34th Symposium on Theoretical Aspects of Computer Science (STACS) (2017), 43, pp. 1–14.

[**c4**] B. Durand, A. Romashchenko. *Quasiperiodicity and Non-computability in Tilings.* Proc. 41st International Symposium on Mathematical Foundations of Computer Science (MFCS) (2015), Part 1, pp. 218–230.

[c5] D. Chumbalov, A. Romashchenko. *Randomized Polynomial Time Protocol for Combinatorial Slepian-Wolf Problem.* Proc. 41st International Symposium on Mathematical Foundations of Computer Science (MFCS) (2015), Part 2, pp. 235–247.

[**c6**] A. Romashchenko. *Pseudo-random graphs and bit probe schemes with one-sided error.* In Proc. 6th International Computer Science Symposium in Russia (CSR). Saint-Petersburg, Russia (2011), Lecture Notes in Computer Science 6651, Springer. pp. 50–63.

[**c7**] B. Durand, A. Romashchenko, A. Shen. *High complexity tilings with sparse errors.* In Proc. of 36th International Colloquium on Automata, Languages, and Pro-

gramming (ICALP) 1, (2009), pp. 403–414.

[**c8**] D. Musatov, A. Romashchenko, A. Shen. *Variations on Muchnik's Conditional Complexity Theorem.* In Proc. 4th International Computer Science Symposium in Russia (CSR). Novosibirsk, Russia, (2009), pp. 250–262.

[**c9**] B. Durand, A. Romashchenko, A. Shen. *Fixed Point and Aperiodic Tilings.* In Proc. 12th International Conference on Developments in Language Theory (DLT). Kyoto, Japan (2008), pp. 537–548.

[**c10**] An.A. Muchnik, A.E. Romashchenko. *A Random Oracle Does Not Help Extract the Mutual Information.* In Proc. 33rd International Symposium Mathematical Foundations of Computer Science (MFCS). Poland, Torun (2008), pp. 527–538.

[c11] A. Romashchenko. *Reliable Computations Based on Locally Decodable Codes.* Proc. 23rd Annual Symposium on Theoretical Aspects of Computer Science (STACS), Marseille, France (2006), pp. 537–548.

[c12] T. Lee, A. Romashchenko. *On Polynomially Time Bounded Symmetry of Information.* Proc. 29th Symposium on the Mathematical Foundations of Computer Science (MFCS). Prague (2004), pp. 463–475.

[**c13**] A. Romashchenko. *Extracting the Mutual Information for a Triple of Binary Strings.* Proc. 18th Annual IEEE Conference on Computational Complexity (CCC), Aarhus, Denmark (2003), pp. 221–229.

[**c14**] A. Romashchenko, A. Shen, N. Vereshchagin. *Combinatorial Interpretation of Kolmogorov Complexity.* Proc. 15th Annual IEEE Conference on Computational Complexity (CCC), Florence, Italy (2000), pp. 131–138.

[c15] An. A. Muchnik, A. Shen, A. Romashchenko, N. Vereshchagin. *Upper semilattice of binary strings with the relation x is simple conditional to y.* Proc. 14th Annual IEEE Conference on Computational Complexity (CCC), Atlanta, Georgia. (1999), pp. 114–122.

[c16] D. Hammer, A. Romashchenko, A. Shen, N. Vereshchagin. *Inequalities for Shannon entropies and Kolmogorov complexities.* Proc. Twelfth Annual IEEE Conference on Computational Complexity (CCC). Ulm, Germany (1997), pp. 13–23.

Actes d'autres colloques et workshops avec comité de lecture

[**c17**] T. Kaced, A. Romashchenko. *On the Non-robustness of Essentially Conditional Information Inequalities.* IEEE Information Theory Workshop (ITW) (2012), pp. 262–266.

[c18] A. Shen, A. Romashchenko. *Topological arguments for Kolmogorov complexity.* In Proc. 18th International Workshop on Cellular Automata and Discrete Complex Systems (AUTOMATA) (2012), pp. 127–132.

[**c19**] B. Durand, A. Romashchenko, A. Shen. *Effective Closed Subshifts in* $1D$ *Can Be Implemented in* $2D$. Fields of Logic and Computation: Essays Dedicated to Yuri Gurevich on the Occasion of His 70th Birthday, Lecture Notes in Computer Science, vol. 300, Springer-Verlag, (2010), pp. 208–226.

[**c20**] T. Kaced, A. Romashchenko. *On essentially conditional information inequalities.* In Proc. IEEE Int. Sympos. on Information Theory (ISIT), (2011), pp. 1935–1939.

[**c21**] A. Shen, L. Bienvenu, A. Romashchenko. *Sparse sets.* In Proceedings of the First Symposium on Cellular Automata 'Journées Automates Cellulaires' (JAC), Uzès, France (2008), pp. 18–28.

[c22] B. Durand, A. Romashchenko. *On stability of computations by cellular automata.* In Proc. European Conf. Compl. Syst., Paris 2005.

Livres

[b1] Alexander Shen, Andrei Romashchenko, Andrey Rumyantsev. *Notes on Coding Theory.* (in Russian) MCCME Publishers, Moscow. 1st ed. 2011, 2nd ed. 2017.

Quelques publications éléctroniques

[**e1**] A. Romashchenko and M. Zimand. *An operational characterization of mutual information in algorithmic information theory.* Electronic Colloquium on Computational Complexity (ECCC) TR18-043 (2018). 36 pp. (une version courte est publiée á ICALP 2018)

[**e2**] B. Durand and A. Romashchenko. *The expressiveness of quasiperiodic and minimal shifts of finite type.* `arXiv:1802.01461` (2018).

General Bibliography

[1] R. V. Hartley, "Transmission of information", *Bell Labs Technical Journal*, vol. 7, no. 3, pp. 535–563, 1928.

[2] C. E. Shannon, "Communication theory of secrecy systems", *Bell Sys. Tech. Jour.*, vol. 28, pp. 623–656, 1948.

[3] A. N. Kolmogorov, "Three approaches to the quantitative definition of information", *Problems Inform. Transmission*, vol. 1, no. 1, pp. 1–7, 1965.

[4] R. Berger, *The undecidability of the domino problem*, 66. American Mathematical Soc., 1966.

[5] F. C. Hennie and R. E. Stearns, "Two-tape simulation of multitape turing machines", *Journal of the ACM*, vol. 13, no. 4, pp. 533–546, 1966.

[6] P. Martin-Löf, "The definition of random sequences", *Information and Control*, vol. 9, pp. 602–619, 1966.

[7] B. H. Bloom, "Space/time trade-offs in hash coding with allowable errors", *Communications of the ACM*, vol. 13, no. 7, pp. 422–426, 1970.

[8] A. Zvonkin and L. Levin, "The complexity of finite objects and the development of the concepts of information and randomness by means of the theory of algorithms", *Russian Mathematical Surveys*, vol. 25, no. 6, pp. 83–124, 1970.

[9] R. M. Robinson, "Undecidability and nonperiodicity for tilings of the plane", *Inventiones mathematicae*, vol. 12, no. 3, pp. 177–209, 1971.

[10] L. Bassalygo and M. Pinsker, "The complexity of an optimal non-blocking commutation scheme without reorganization", *Problems of Information Transmission*, vol. 9, no. 1, pp. 84–87, 1973.

[11] P. Gács and J. Körner, "Common information is far less than mutual information", *Probl. Control Inf. Theory*, vol. 2, no. 2, pp. 149–162, 1973.

[12] M. S. Pinsker, "On the complexity of a concentrator", in *7th International Telegraffic Conference*, Citeseer, vol. 4, 1973, pp. 1–318.

[13] D. Slepian and J. Wolf, "Noiseless coding of correlated information sources", *IEEE Transactions on Information Theory*, vol. 19, no. 4, pp. 471–480, 1973.

[14] R. Ahlswede and J. Körner, "On common information and related characteristics of correlated information sources", in *Preprint. Presented at the 7th Prague Conference on Information Theory*, 1974.

[15] W. Hanf, "Nonrecursive tilings of the plane. I", *The Journal of Symbolic Logic*, vol. 39, no. 2, pp. 283–285, 1974.

[16] D. Myers, "Nonrecursive tilings of the plane. II", *The Journal of Symbolic Logic*, vol. 39, no. 2, pp. 286–294, 1974.

[17] R. Ahlswede and J. Korner, "Source coding with side information and a converse for degraded broadcast channels", *IEEE Transactions on Information Theory*, vol. 21, no. 6, pp. 629–637, 1975.

[18] G. J. Chaitin, "A theory of program size formally identical to information theory", *Journal of the ACM*, vol. 22, no. 3, pp. 329–340, 1975.

[19] A. N. Kolmogorov, "Talk at the seminar at moscow state university mathematics department (logic division).", Nov. 1981.

[20] P. van Emde Boas, *Dominoes are forever*. Universiteit van Amsterdam, Mathematisch Instituut, 1983.

[21] A. Kh. Shen, "The concept of (α, β)-stochasticity in the Kolmogorov sense, and its properties", in *Soviet Math. Dokl*, vol. 28, 1983, pp. 295–299.

[22] M. L. Fredman, J. Komlós, and E. Szemerédi, "Storing a sparse table with 0 (1) worst case access time", *Journal of the ACM*, vol. 31, no. 3, pp. 538–544, 1984.

[23] F. R. Chung, R. L. Graham, P. Frankl, and J. B. Shearer, "Some intersection theorems for ordered sets and graphs", *Journal of Combinatorial Theory, Series A*, vol. 43, no. 1, pp. 23–37, 1986.

[24] P. Gács, "Reliable computation with cellular automata", *Journal of Computer and System Sciences*, vol. 32, no. 1, pp. 15–78, 1986.

[25] B. Grünbaum and G. C. Shephard, *Tilings and patterns*. Freeman, 1987.

[26] A. Fiat, M. Naor, J. Schmidt, and A. Siegel, "Non-oblivious hashing", in *Proceedings of the twentieth annual ACM symposium on Theory of computing*, ACM, 1988, pp. 367–376.

[27] P. Gács, *Lecture notes on descriptional complexity and randomness*. Boston University, Graduate School of Arts and Sciences, Computer Science Department, 1988.

[28] L. S. Levitov, "Local rules for quasicrystals", *Communications in mathematical physics*, vol. 119, no. 4, pp. 627–666, 1988.

[29] N. Nisan, "Pseudorandom generators for space-bounded computation", *Combinatorica*, vol. 12, no. 4, pp. 449–461, 1992.

[30] R. Ahlswede and I. Csiszár, "Common randomness in information theory and cryptography - I: secret sharing", *IEEE Trans. Information Theory*, vol. 39, no. 4, pp. 1121–1132, 1993.

[31] U. M. Maurer, "Secret key agreement by public discussion from common information", *IEEE Trans. Information Theory*, vol. 39, no. 3, pp. 733–742, 1993.

[32] N. Nisan and A. Wigderson, "Hardness vs randomness", *Journal of computer and System Sciences*, vol. 49, no. 2, pp. 149–167, 1994.

[33] B. Bollobás and A. Thomason, "Projections of bodies and hereditary properties of hypergraphs", *Bulletin of the London Mathematical Society*, vol. 27, no. 5, pp. 417–424, 1995.

[34] N. Kahale, "Eigenvalues and expansion of regular graphs", *Journal of the ACM*, vol. 42, no. 5, pp. 1091–1106, 1995.

[35] F. Matúš, "Conditional independences among four random variables ii", *Combinatorics, Probability and Computing*, vol. 4, no. 4, pp. 407–417, 1995.

[36] K. Culik II, "An aperiodic set of 13 wang tiles", *Discrete Mathematics*, vol. 160, no. 1-3, pp. 245–251, 1996.

[37] J. Kari, "A small aperiodic set of wang tiles", *Discrete Mathematics*, vol. 160, no. 1-3, pp. 259–264, 1996.

[38] J. Von Neumann and A. W. Burks, *Theory of self-reproducing automata*. University of Illinois Press Urbana, 1996.

[39] C. Allauzen and B. Durand, "Tiling problems", *E. Borger, E. Gradel, Yu. Gurevich, The classical decision problem. Springer-Verlag*, 1997.

[40] E. Kushilevitz and N. Nisan, *Communication complexity*. Cambridge, 1997.

[41] Z. Zhang and R. W. Yeung, "A non-Shannon-type conditional inequality of information quantities", *IEEE Transactions on Information Theory*, vol. 43, no. 6, pp. 1982–1986, 1997.

[42] C. H. Bennett, P. Gács, M. Li, P. M. Vitányi, and W. H. Zurek, "Information distance", *IEEE Transactions on information theory*, vol. 44, no. 4, pp. 1407–1423, 1998.

[43] An. A. Muchnik, "On common information", *Theor. Comput. Sci.*, vol. 207, pp. 319–328, 1998.

[44] Z. Zhang and R. W. Yeung, "On characterization of entropy function via information inequalities", *IEEE Transactions on Information Theory*, vol. 44, no. 4, pp. 1440–1452, 1998.

[45] A. Brodnik and J. I. Munro, "Membership in constant time and almost-minimum space", *SIAM Journal on Computing*, vol. 28, no. 5, pp. 1627–1640, 1999.

[46] B. Durand, "Tilings and quasiperiodicity", *Theoretical Computer Science*, vol. 221, no. 1-2, pp. 61–75, 1999.

[47] L. Trevisan, "Constructions of near-optimal extractors using pseudo-random generators", in *Proceedings of the 30th ACM Symposium on Theory of Computing*, ACM Press, May 1999, pp. 141–148.

[48] J. Radhakrishnan and A. Ta-Shma, "Bounds for dispersers, extractors, and depth-two superconcentrators", *SIAM Journal on Discrete Mathematics*, vol. 13, no. 1, pp. 2–24, 2000.

[49] H. Buhrman, L. Fortnow, and S. Laplante, "Resource-bounded Kolmogorov complexity revisited", *SIAM Journal on Computing*, vol. 31, no. 3, pp. 887–905, 2001.

[50] L. F. Gray, “A reader’s guide to gacs’s “positive rates” paper”, *Journal of Statistical Physics*, vol. 103, no. 1-2, pp. 1–44, 2001.

[51] R. Pagh, “Low redundancy in static dictionaries with constant query time”, *SIAM Journal on Computing*, vol. 31, no. 2, pp. 353–363, 2001.

[52] ——, “On the cell probe complexity of membership and perfect hashing”, in *Proceedings of the thirty-third annual ACM symposium on Theory of computing*, ACM, 2001, pp. 425–432.

[53] A. Ta-Shma, C. Umans, and D. Zuckerman, “Loss-less condensers, unbalanced expanders, and extractors”, in *Proceedings of the thirty-third annual ACM symposium on Theory of computing*, ACM, 2001, pp. 143–152.

[54] H. Buhrman, P. B. Miltersen, J. Radhakrishnan, and S. Venkatesh, “Are bitvectors optimal?”, *SIAM Journal on Computing*, vol. 31, no. 6, pp. 1723–1744, 2002.

[55] An. A. Muchnik, “Conditional complexity and codes”, *Theor. Comput. Sci.*, vol. 271, no. 1-2, pp. 97–109, 2002.

[56] A. Ta-Shma, “Storing information with extractors”, *Information Processing Letters*, vol. 83, no. 5, pp. 267–274, 2002.

[57] L. Devroye and P. Morin, “Cuckoo hashing: further analysis”, *Information Processing Letters*, vol. 86, no. 4, pp. 215–219, 2003.

[58] Y. Horibe, “A note on Kolmogorov complexity and entropy”, *Applied mathematics letters*, vol. 16, no. 7, pp. 1129–1130, 2003.

[59] I. Csiszár and P. Narayan, “Secrecy capacities for multiple terminals”, *IEEE Trans. Information Theory*, vol. 50, no. 12, pp. 3047–3061, 2004.

[60] R. Pagh and F. F. Rodler, “Cuckoo hashing”, *Journal of Algorithms*, vol. 51, no. 2, pp. 122–144, 2004.

[61] H. Buhrman, T. Lee, and D. van Melkebeek, “Language compression and pseudorandom generators”, *Computational Complexity*, vol. 14, no. 3, pp. 228–255, 2005.

[62] B. Durand, L. Levin, and A. Shen, “Local rules and global order, or aperiodic tilings”, *The Mathematical Intelligencer*, vol. 27, no. 1, pp. 64–68, 2005.

[63] L. A. Levin, “Aperiodic tilings: breaking translational symmetry”, *The Computer Journal*, vol. 48, no. 6, pp. 642–645, 2005.

[64] R. Ahlswede and J. Körner, “On common information and related characteristics of correlated information sources”, in *General Theory of Information Transfer and Combinatorics*, Springer, 2006, pp. 664–677.

[65] T. M. Cover and J. A. Thomas, *Elements of information theory (2. ed.)* Wiley, 2006.

[66] S. Hoory, N. Linial, and A. Wigderson, “Expander graphs and their applications”, *Bulletin of the American Mathematical Society*, vol. 43, no. 4, pp. 439–561, 2006.

[67] O. Reingold, R. Shaltiel, and A. Wigderson, "Extracting randomness via repeated condensing", *SIAM Journal on Computing*, vol. 35, no. 5, pp. 1185–1209, 2006.

[68] A. Yu. Rumyantsev and M. A. Ushakov, "Forbidden substrings, Kolmogorov complexity and almost periodic sequences", in *Annual Symposium on Theoretical Aspects of Computer Science*, Springer, 2006, pp. 396–407.

[69] N. Alon, I. Newman, A. Shen, G. Tardos, and N. Vereshchagin, "Partitioning multi-dimensional sets in a small number of "uniform" parts", *European Journal of Combinatorics*, vol. 28, no. 1, pp. 134–144, 2007.

[70] L. Antunes, S. Laplante, A. Pinto, and L. Salvador, "Cryptographic security of individual instances", in *International Conference on Information Theoretic Security*, Springer, 2007, pp. 195–210.

[71] F. Matus, "Infinitely many information inequalities", in *Information Theory, 2007. ISIT 2007. IEEE International Symposium on*, IEEE, 2007, pp. 41–44.

[72] F. Matúš, "Adhesivity of polymatroids", *Discrete Mathematics*, vol. 307, no. 21, pp. 2464–2477, 2007.

[73] A. D. Smith, "Scrambling adversarial errors using few random bits, optimal information reconciliation, and better private codes", in *Proceedings of the eighteenth annual ACM-SIAM Symposium on Discrete algorithms*, Citeseer, vol. 7, 2007, pp. 395–404.

[74] B. Durand, L. A. Levin, and A. Shen, "Complex tilings", *The Journal of Symbolic Logic*, vol. 73, no. 2, pp. 593–613, 2008.

[75] ——, "Complex tilings", *The Journal of Symbolic Logic*, vol. 73, no. 2, pp. 593–613, 2008.

[76] M. Li and P. Vitanyi, *An introduction to Kolmogorov complexity and its applications*. Springer-Verlag, 2008, 3rd edition. 1st edition in 1993.

[77] N. Ollinger, "Two-by-two substitution systems and the undecidability of the domino problem", in *Conference on Computability in Europe*, Springer, 2008, pp. 476–485.

[78] A. Ballier, "Propriétés structurelles, combinatoires et logiques des pavages", PhD thesis, Université de Marseille, 2009.

[79] V. Guruswami, C. Umans, and S. P. Vadhan, "Unbalanced expanders and randomness extractors from Parvaresh–Vardy codes", *Journal of the ACM*, vol. 56, no. 4, 2009.

[80] M. Hochman, "On the dynamics and recursive properties of multidimensional symbolic systems", *Inventiones mathematicae*, vol. 176, no. 1, p. 131, 2009.

[81] A. Ballier and E. Jeandel, "Computing (or not) quasi-periodicity functions of tilings", *arXiv preprint arXiv:1012.1222*, 2010.

[82] M. Hochman and T. Meyerovitch, "A characterization of the entropies of multidimensional shifts of finite type", *Annals of Mathematics*, pp. 2011–2038, 2010.

[83] M. Braverman, "Poly-logarithmic independence fools bounded-depth boolean circuits", *Communications of the ACM*, vol. 54, no. 4, pp. 108–115, 2011.

[84] T. Chan, "Recent progresses in characterising information inequalities", *Entropy*, vol. 13, no. 2, pp. 379–401, 2011.

[85] I. Csiszar and J. Körner, *Information theory: coding theorems for discrete memoryless systems*. Cambridge University Press, 2011.

[86] I. Razenshteyn, "Common information revisited", *arXiv preprint arXiv:1104.3207*, 2011.

[87] R. W. Yeung, *A first course in information theory*. Springer Science & Business Media, 2012.

[88] N. Aubrun and M. Sablik, "Simulation of effective subshifts by two-dimensional subshifts of finite type", *Acta applicandae mathematicae*, vol. 126, no. 1, pp. 35–63, 2013.

[89] E. Jeandel and P. Vanier, "Turing degrees of multidimensional sfts", *Theoretical Computer Science*, vol. 505, pp. 81–92, 2013.

[90] T. Kaced, "Equivalence of two proof techniques for non-Shannon-type inequalities", in *Proceedings of IEEE International Symposium on Information Theory (ISIT)*, IEEE, 2013, pp. 236–240.

[91] B. Bauwens and M. Zimand, "Linear list-approximation for short programs (or the power of a few random bits)", in *Proceedings of the 29th IEEE Conference on Computational Complexity*, 2014, pp. 241–247.

[92] L. Csirmaz, "Book inequalities", *IEEE Transactions on Information Theory*, vol. 60, no. 11, pp. 6811–6818, 2014.

[93] D. Musatov, "Improving the space-bounded version of Muchnik's conditional complexity theorem via "naive" derandomization", *Theory of computing systems*, vol. 55, no. 2, pp. 299–312, 2014.

[94] E. Jeandel and M. Rao, "An aperiodic set of 11 wang tiles", *arXiv preprint arXiv:1506.06492*, 2015.

[95] C. Zinoviadis, "Hierarchy and expansiveness in 2D subshifts of finite type", in *International Conference on Language and Automata Theory and Applications*, Springer, 2015, pp. 365–377.

[96] C. Goodman-Strauss, "Lots of aperiodic sets of tiles", *arXiv preprint arXiv:1608.07165*, 2016.

[97] F. Matúš and L. Csirmaz, "Entropy region and convolution", *IEEE Transactions on Information Theory*, vol. 62, no. 11, pp. 6007–6018, 2016.

[98] C. Zinoviadis, "Hierarchy and expansiveness in two-dimensional subshifts of finite type", *arXiv preprint arXiv:1603.05464*, 2016.

[99] A. Shen, V. Uspensky, and N. Vereshchagin, *Kolmogorov complexity and algorithmic randomness*. American Mathematical Society, 2017.

[100] N. Vereshchagin and A. Shen, "Algorithmic statistics: forty years later", in *Computability and Complexity*, Springer, 2017, pp. 669–737.

[101] L. B. Westrick, "Seas of squares with sizes from a Π^0_1 set", *Israel Journal of Mathematics*, vol. 222, no. 1, pp. 431–462, 2017.

[102] M. Zimand, "Kolmogorov complexity version of Slepian-Wolf coding", in *Proceedings of the annual ACM Symposium on Theory of Computing*, ACM, Jun. 2017, pp. 22–32.

[103] M. Zimand, "Distributed compression through the lens of algorithmic information theory: a primer", *arXiv preprint arXiv:1706.08468*, 2017.

[104] B. Bauwens, A. Makhlin, N. Vereshchagin, and M. Zimand, "Short lists with short programs in short time", *Computational complexity*, vol. 27, no. 1, pp. 31–61, 2018.