MATEMATICKO-FYZIKÁLNÍ
FAKULTA
Univerzita Karlova

BAKALÁŘSKÁ PRÁCE

Michal Sekerka

Kolmogorovovská složitost a Shannonova informace

Informatický ústav Univerzity Karlovy

Vedoucí bakalářské práce:	prof. Mgr. Michal Koucký, Ph.D.
Studijní program:	Matematické metody informační bezpečnosti
Studijní obor:	Mathematical Methods of Information Security

Praha 2019

Prohlašuji, že jsem tuto bakalářskou práci vypracoval(a) samostatně a výhradně s použitím citovaných pramenů, literatury a dalších odborných zdrojů. Tato práce nebyla využita k získání jiného nebo stejného titulu.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona v platném znění, zejména skutečnost, že Univerzita Karlova má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle §60 odst. 1 autorského zákona.

V .... dne ....

Podpis autora

Poděkování. V první řadě bych na tomto místě rád poděkoval svým rodičům za jejich neutuchající podporu, kterou projevovali po celou dobu mého studia. Dále bych chtěl vyjádřit svůj vděk panu profesoru Kouckému, za to, že souhlasil s vedením této práce, nechal mi volné pole působnosti a vždy byl ochoten si na mě udělat čas k prodiskutování vznikajícího textu. V neposlední řadě děkuji panu profesoru Hlubinkovi za poskytnutí reference k teorii pravděpodobnosti a panu profesoru Žemličkovi, který se i přes délku textu ujal oponentury této práce.

Název práce: Kolmogorovovská složitost a Shannonova informace

Autor: Michal Sekerka

Katedra algebry: Informatický ústav Univerzity Karlovy

Vedoucí bakalářské práce: prof. Mgr. Michal Koucký, Ph.D., Informatický ústav Univerzity Karlovy

Abstrakt: V průběhu dvacátého století vznikly dvě úspěšné formalizace kvantitativního aspektu informace spojeným s komunikací zprávy: Shannonova informace a Kolmogorovovská složitost. Obě zmíněné definice vyplývají ze dvou separátních odvětví matematiky. Shannonova informace vznikla jako aplikace elementární teorie pravděpodobnosti a statistiky. Je definovaná jako funkce v pravděpodobnosti s tím, že určuje jakýsi spodní odhad na binární kompresi. Kolmogorovovská složitost má na druhou stranu kořeny ve formální logice a teorii řešitelnosti. Jedná se o délku minimálního algoritmického popisu zprávy. Je překrásným důsledkem, že za určitých podmínek tyto dvě veličiny vycházejí až na zanedbatelnou chybu asymptoticky stejně. Moje práce má za úkol formálně zavést obě veličiny, porovnat jejich nedostatky, zaměřit se na jejich podobnosti a rozdíly a v neposlední řadě dokázat jejich kýžený vztah.

Klíčová slova: Kolmogorovovská složitost Shannonova informace Turingovy stroje Entscheidungsproblem algoritmická řešitelnost binární komprese

Title: Kolmogorov complexity and Shannon information

Author: Michal Sekerka

Department of Algebra : Computer Science Institute of Charles University

Supervisor: prof. Mgr. Michal Koucký, Ph.D., Informatický ústav Univerzity Karlovy

Abstract: There arose two successful formalisations of the quantitative aspect of information over the course of the twentieth century: Shannon information and Kolmogorov complexity. Both afore mentioned definitions are rooted in mostly separate parts of mathematics. Shannon's information came to existence as an application of the elementary theory of probability and statistics. It is defined as a function in probability, with it being a lower bound on binary compression. Kolmogorov complexity, on the other hand, springs from formal logic and theory of computability. Kolmogrov defined it as the length of a minimal algorithmic description of a message. It is a beautiful result that when certain conditions do apply then those two functions behave asymptotically equivalently. My thesis is concerned with formally defining both measures of information, comparing their drawbacks, highlighting their similarities and differences and at last but not least proving their coveted asymptotic relationship.

Keywords: Kolmogorov complexity Shannon information Turing machines Entscheidungsproblem computability binary compression

Obsah

1. Úvod

Informace se v poslední době těší nevídaně velké pozornosti jak ve vědním tak i ve veřejném kontextu. Díky rostoucímu vlivu informace a všech aspektů její výměny na politický, ekonomický a všeobecně sociální vývoj, dokonce někteří lidé označují dobu, ve které žijeme informačním věkem (Wikipedia contributors, 2019a). Přesto má jen málokdo jasnou představu o tom, co tento pojem ve skutečnosti znamená. Koncept informace má kořeny v antické filosofii. Konkrétně pochází z prací starořeckého filosofa Platóna. Jeho celoživotní dílo, kterým velice významně přispěl do dějin filosofie, je dnes široce citované pod názvem teorie forem. Ta je spojena asi s nejznámějším podobenstvím filosofie, známým jako Platónova jeskyně. Platónova jeskyně byla sepsána v Sedmé knize republiky jako sokratovský dialog mezi Sokratem a Platónovým bratrem Glauconem. Dobře ilustruje oč v teorii forem jde: (*převyprávěno, plná verze např.* (Hamilton a Cairns, 1992))

Představte si muže, kteří jsou od narození drženi v temné podzemní jeskyni. Jejich nohy a krk jsou spoutány tak, že jsou nuceni dívat se na prázdnou stěnu této jeskyně, neschopni rozhlédnout se okolo sebe. Za nimi žhne plamen, který je osvětluje a promítá jejich stíny na onu stěnu. Před tímto ohněm čas od času prochází lidé nosící loutky a jiné předměty. Plamen vrhá stíny těchto předmětů na stěnu sledovanou vězni. Z venku se navíc ozývají ozvěny zvuků linoucí se dolů schodištěm dovnitř jeskyně až ke spoutaným mužům. Neschopni podívat se za sebe a neznaje svět okolo nich jsou vězni přesvědčeni, že stíny jsou vším, co existuje. Pojmenovávají je a přiřazují jim zvuky, které slyší zvenčí. Pro vězně jsou stíny realitou. Sokratés následně Glauconovi popisuje příběh vězně jehož pouta jsou zlomena a který je násilně vyvlečen na svobodu. Když se dostane ven na slunce, tak se zprvu není schopen přizpůsobit novému prostředí. Z venkovního světa je rozčílen a hledá útěchu ve stínech okolních objektů. Ale nakonec, když si jeho oči zvyknou, tak mu stíny začnou připadat oproti krásné rozličnosti širého okolí nudné a nezajímavé. Sokratés potom líčí, jak se tento vězeň vrátí zpět do jeskyně a nadšeně vypráví ostatním vězňům o neuvěřitelném světě tam venku. Ostatní vězni si myslí, že se při své cestě ven zbláznil a brání se tomu, aby je odtáhnul na svobodu. Když ovšem nepřestává naléhat, tak jej ve strachu zabijí. (Tím se Platón očividně odkazuje na život jeho učitele Sokrata, který byl za rušení veřejného pořádku v Aténách popraven.)

Toto neuvěřitelně stimulující a barvité podobenství dobře ukazuje základní myšlenku Platónovy filosofie. Dle Platóna všechny předměty, které vnímáme v reálném světě nejsou ničím jiným než nedokonalým ztělesněním jejich idealizované formy. Tu Platón nazývá eidós (=$\epsilon\tilde{\iota}\delta o\varsigma$). Eidós tedy znamená nějakou idealizovanou identitu předmětu, to čím doopravdy je zbavíme-li jej všeho jeho nadbytečného fyzického tvaru.(The Editors of Encyclopaedia Britannica, 2011) Podle Platóna cokoliv, co vnímáme je pouhým odrazem této esence ne nepodobným stínu na stěně. Římští filosofové navazující na Platóna a Aristotela (např. Cicero a Augustin) zaměňovali řecké slovo eidós za latinské slovo informare, ze kterého je očividně odvozené slovo informace. (Adriaans, 2019) Je velice zajímavé, že anglická slova idea (=eidós) neboli myšlenka a information (=informare) znamenající informace mají společnou etymologii. Z dnešního užití těchto slov je ovšem cítit, že se za dobu používání jejich význam od tohoto kořene oddálil. Informace se dnes používá v podstatě výhradně v kontextu nějaké komunikace. Centrální téma je ovšem stále stejné. Informace spojená s nějakým sdělením je jeho ideální identitou. Tím čím doopravdy je, zbavíme-li jej všeho jeho „syntaktického“ (nadbytečnost ve formulaci, jazyce, písmu,...) a fyzického (médium přenosu, komunikační kanál,...) tvaru. Odstraníme-li tedy ze zprávy všechnu její redundanci, tak to, co nám zbyde, ona čistá koncentrovaná esence této zprávy, je informací, kterou přenáší. Ze zmíněné intuice je poměrně silně cítit vztah informace a komprese. Ten bude nesmírně důležitý pro zbytek naší práce.

Posun v chápání pojmu informace od jeho filosofického kontextu po to, jak je používán dnes, nebyl bohužel přímočarý a jeho výčet by vydal na samostatnou práci. Popisovat jej zde tedy nebudeme. Zmiňme však, že nejznatelnější změna se udála ve dvacátém století. Přičemž v jeho polovině byla informace již vědeckou komunitou v podstatě výhradně používána ve smyslu nějaké měřitelné abstraktní veličiny schopné nést význam. Jako hybatele této změny uveďme britského statistika R.A. Fishera, který roku 1925 jako první použil pojem informace v matematickém textu (Oxford English Dictionary Contributors, 2009) a významně tak přispěl ke změně jejího tehdejšího významu:

> *„To, o čem jsme mluvili jako o chybové křivce, může být stejně tak vnímáno jako množství informace v jediném pozorování příslušném danému rozdělení.“*

Již jsme zmínili, že informace je pojem, poměrně hluboce zakořeněný ve vědě. Různé vědy si však už ve dvacátém století začínaly na informaci vytvářet rozdílné pohledy. Přičemž mezi některými z nich je dodnes jen velice těžké hledat paralely. Jinou interpretaci tohoto pojmu bude pravděpodobně mít biolog ve vztahu ke genetické informaci, lingvista k informaci napsané v literárním díle, nebo fyzik při přemítání nad termodynamickou informací. Dává vůbec smysl porovnávat kolik informace je uloženo v DNA a kolik je napsáno v Shakespearově Hamletovi? Do poloviny dvacátého století se to zdálo nemožné. Informace byla chápána jako spíše filosofický a obecně chabě definovaný pojem. Zlomovým bodem bylo, když roku 1948 americký elektroinženýr Claude Shannon vydal článek Mathematical Theory of Communication (Shannon, 1948). Shannon si uvědomil, že chcete-li měřit informaci nějaké zprávy a zbavíte-li ji veškeré její formy, tak to, co vám zbyde, je koncept pravděpodobnosti. Ta v Shannonově teorii vystihuje míru překvapení ve chvíli, kdy danou zprávu obdržíme. Shannon tím ukázal, že přistoupíme-li na některé pravděpodobnostní předpoklady, tak nejenže informace existuje, ale jako jakákoliv jiná fyzikální veličina je v praxi měřitelná a výsledky takového měření jsou naproti sobě porovnatelné.

S postupem času začínala být čím dál zřetelnější užitečnost Shannonovy práce. Kromě toho, že pomohla vyřešit některé komunikační problémy spojené s nespolehlivostí informačního kanálu (ke kterým byla určena), tak nabídla matematickou optiku k interpretaci kvantitativního aspektu informace. Ve fyzice například významným způsobem posunula debatu nad některými paradoxy, jako byl Maxwellův démon a dala nový pohled na vztah informace a energie zastoupeném například v konceptech Szilardova motoru a Landauerova principu. Shannon takto v podstatě jedním článkem nastartoval intelektuální revoluci. Informace se od té doby stala nedílnou součástí většiny věd.

I přes to všechno není žádným tajemstvím, že Shannonova teorie zdědila některé koncepční nedostatky z teorie pravděpodobnosti. Sovětský matematik Andrej Nikolajevič Kolmogorov nabídl (Kolmogorov, 1968) (mimo jiné) k vyplnění mezery, kterou tyto nedostatky zanechaly alternativní pohled na míru informace. Kolmogorov byl informační teorií fascinován. Viděl v ní totiž způsob, jak vyřešit určité problémy aplikovatelosti axiomatické teorie pravděpodobnosti (Porter, 2014). K tomuto účelu ovšem bylo potřeba sprostit definici informace závislosti na hodnotě pravděpodobnosti. Definice, kterou navrhl, proto měřila míru vnitřní algoritmické struktury zprávy. Tato definice je dnes známá pod pojmem Kolmogorova složitost. Našla široké uplatnění v odvětvích, ve kterých je výhodnější vnímat zprávu spíše izolovaně než v rámci nějakého globálního kontextu. Dá se s ní setkat všude od strojového učení, algoritmické statistiky, až k formalizaci zdánlivě nesouvisejících konceptů náhody a Occamovy břitvy. Dobrý přehled všech aplikací lze nalézt například v knize (Li a Vitányi, 2008).

Cílem mé práce je formálně zavést obě veličiny včetně alternativního pohledu na asymptotiku Shannonovy informace a jejího vztahu ke kompresi. Ukážu jejich výhody a nedostatky a v neposlední řadě dokážu jejich asymptotický vztah. Vztah Shannonovy a Kolmogorovovy teorie je dle mého názoru jedním z nejhezčích důsledků matematiky. Ukážeme v něm, že délka algoritmického popisu zprávy a Shannonova míra informace asymptoticky splývají a nalezneme tak souvislost mezi dvěma v podstatě separátními odvětvími matematiky.

Při zavádění Kolmogorovovské složitosti se nevyhneme úvodu do algoritmické řešitelnosti. Jako model výpočtu jsme zvolili Turingovy stroje. V rámci této části mé práce prezentuji konstrukci univerzálního Turingova stroje a snažím se zde vyplnit některé formální mezery, které jsou v literatuře při konstrukci strojů obvykle ponechány. Pro zdůraznění role deskripčního Turingova čísla jako popisu Turingova stroje jsem zavedl pojem isomorfismu Turingových strojů. Z koncepčních důvodů tuto sekci zakončuji formálním důkazem nerozhodnutelnosti logiky prvního řádu a problémem rozhodnutí.

Všechny důležité definice jsou v textu odůvodněné a pokud je třeba, tak je k nim uveden krátký historický kontext.

1.1 Opomenutí významu

Jistý aspekt informace tak, jak je pojata Shannonem a Kolmogorovovem může být matoucí. Rád bych jej tedy vzpomenul dříve než se pustíme do formální teorie. Potom, co Shannon vydal práci zabývající se komunikací po informačním kanále, nastal obrovský boom v propagaci a reinterpretaci této teorie. Někdy tomu tak bylo až do bizarních rozměrů. Roku 1956 se v Londýně konalo třetí informační sympózium, ve kterém se objevila aplikace informační teorie v tak rozličných oborech jako byla lingvistika, psychologie, neurofysiologie, fonetika, politologie, atp. Mnoho článků, které v této době vznikaly (často v časopisu IRE) navíc byly jen povrchní reinterpretací Shannonovy teorie do v podstatě nesouvisejících oborů. (Aftab a kol., 2001)

Shannon se k tomuto fenoménu vyjádřil ve článku s názvem „The Bandwagon“ (Shannon, 1956). Upozornil v něm, že informační teorie jím byla formulována pouze za účelem využití v inženýrské praxi a varoval, že při aplikacích v jiných oborech by se mělo přistupovat s nejvyšší opatrností. Shannon se sám nechal slyšet, že se mu zdálo nepravděpodobné, že by kdy mohl existovat koncept informace, který by vystihl celou šířku kontextu, ve kterém se daný pojem užíval. (Shannon, 1950)

Přestože se Shannonova teorie setkala s vysokou mírou nadšení, našla se menší skupina akademiků, kteří tuto teorii kritizovali už v podstatě od samého začátku. Jedním z hlavních představitelů byl Donald McKay. Ten na osmé Macyovské konferenci kritizoval Shannonovu teorii pro opomenutí sémantiky. Ve zkratce měla podle něj být informace uložená ve zprávě chápána spíše ve smyslu toho, jaký význam nese pro příjemce, než aby byla omezována, na to jakým signálem je po informačním kanále komunikována. To neznamená, že Shannonovu kvantitativní definici úplně zavrhnul. Označil ji však za pouhý jeden z informačních aspektů. Ten nazval selekční (dnes někdy také syntaktickou) informací. Zmíněný název určil na základě faktu, že je v Shannonově definici míra informace dána výběrem zprávy z nějaké větší množiny. McKay a řada dalších akademiků jako Gregory Bateson, se snažili přijít s teorií informace, která by v sobě zahrnovala význam. Podobných prací vznikla celá řada. Dané teorie však nikdy nedosáhly ve vědecké komunitě nějakého širšího uznání. Shannonova teorie nad nimi měla vždy navrch pro to, jak jednoduše byla v praxi odhadnutelná, interpretovatelná, aplikovatelná a pro svou nepopiratelnou matematickou eleganci. (Logan, 2012)

Shannon se proti výše zmíněné kritice nijak nebránil. Byl si totiž vědom, že jeho definice nutně nekorespondovala s tím, jak je daný pojem v běžné komunikaci používán. Již ze článku Mathematical Theory of Communication lze však vytušit, že volba oprostit informaci od jejího významu byla ze Shannonovy strany vědomá a cílená. Hned na první stránce Shannon totiž píše: „Sémantické aspekty informace jsou irelevantní vůči inženýrským aplikacím.“ (Shannon, 1948) Nejedná se tedy o náhodu, ale o fundamentální část Shannonovy práce. Je nemožné popsat o jak silný předpoklad se v té době jednalo. Odtržení informace od její sémantiky se do té doby zdálo nemožné a bizarní. Je to však právě koncept informace oddělený od jejího významu, který Shannonovi umožnil dokázat obecná tvrzení týkajících se komunikace po informačním kanále a zajistil tak všeobecný úspěch této teorie. Při intepretaci Shannonovy teorie na každodenní problémy ovšem člověk z tohoto důvodu narazí na řadu paradoxů. V Shanonově teorii například nejvíce informace nese náhodný šum písmen. To se však zdá být poměrně neintuitivní. Těžko bychom asi v běžném kontextu náhodnému šumu přiřadili nějaký význam. Může tedy informace existovat bez jakéhokoliv významu? Racionál lidí, kteří tento postoj zastávají, bývá obvykle následující. Pokud je před vámi kniha formulována nějakým mrtvým jazykem, napsaná písmem, které již není nikdo schopen přečíst, potom tomuto textu ani není nikdo schopen přiřadit nějaký význam. To ovšem neznamená, že v dané knize žádná informace uložená není. Význam je něco, co vyvstane, ve chvíli, kdy informaci interpretujete. Jinými slovy informace není ani tak spojená s významem jako spíš s potenciálem nějaký význam nést. Není tedy závislá na dodatečné interpretaci.

V případě náhodného šumu mu jeho statistické vlastnosti (nepředvídatelnost) v jistým smyslu umožňují nad nějakým hypotetickým ideálním komunikační protokolem přenášet velké množství významu. Proto z tohoto pohledu nese notnou dávku informace. (více v následující kapitole)

Tento postoj má samozřejmě řadu oponentů. Úplně stejně se dá totiž argumentovat, že v podstatě cokoliv v přírodě může nést informaci. Nemá však smysl o informaci v daném kontextu mluvit bez někoho, kdo by k ní byl schopen nějaký význam přiřadit.

V podstatě kdekoliv se téma Shannonovy informace objeví, tak je předmět opomenutí sémantiky dodnes žhavě debatovaný (bez existence nějakého konečného konsensu). Závěrem tedy řekněme následující. Je jasné, že pod pojmem informace se běžně zahrnuje jak její syntaktický, tak její sémantický aspekt. Sémantika se zatím však (úspěšné) matematické formalizaci brání. V naší práci se tedy budeme zabývat výhradně kvantifikací syntaktické informace.

1.2 Literatura a Inspirace

V případech, kdy jsem na určitých místech v textu udělal nějaká historická tvrzení, jsem se snažil uvést buď hned u toho či na konci příslušného odstavce zdroj. Myslím si, že je ovšem vhodné na jednom místě sepsat, z jaké literatury jsem ve které sekci čerpal včetně toho, která část matematiky je odkud převzata či inspirována. Práce na podobné téma byla sepsána Peterem Grünwaldem a Paulem Vitányim (Grünwald, 2003). Ač se moje práce od jejich článku v podstatě ve všem odlišuje, tak byl jednou z mých hlavních inspirací pro sepsání této práce. Tedy domnívám se, že je vhodné jej zde zmínit. Ostatní práce na toto téma se obvykle drží podobné struktury jako zmiňovaný článek. Dle mé znalosti v české literatuře žádná práce na toto téma nevznikla.

- **Úvod** (kapitoly 1.1 a 1.2) V úvodu jsme se pokusil shrnou historickou proměnu pojmu informace. Začal jsem jeho historickými kořeny. Při tom jsem čerpal hlavně ze vzdělávací stránky Standfordské univerzity (Adriaans, 2019), článku Marka Burgina: Ideas of Plato in the Context of Contemporary Science and thematics (Burgin, 2018), vzdělávajících videí bývalého tutora na universitě v Oxfordu Tima Wilsona, Plato's cave analysis (Wilson, 2011) a ze článku v encyklopedii Britanice (The Editors of Encyclopaedia Britannica, 2011). V podstatě všechna informace z textu, ve kterém popisuju historii spojenou s Shannonvou teorií, je dohledatelná ve článcích: What Is Information?: Why Is It Relativistic and What Is Its Relationship to Materiality, Meaning and Organization od Robert K. Logana (Logan, 2012) a Information Theory and The Digital Age od autorů Aftaba, Cheunga, Kima, Thakkara a Yeddanapudiho (Aftab a kol., 2001).

- **Shannonova teorie** (celá kapitola 2) Vzhledem k tomu, že historie Shannonovy teorie byla nastíněna v úvodní kapitole, tak v druhé kapitole popisuji čistě matematiku Shannonovy informace. Tvrzení 2.1.1, 2.1.2 jsou pozměněná a dotáhnutá do konce ze vzdělávacího textu Toma Cartera (Carter, str. 15 - 16). Tvrzení 2.1.5-2.1.7 jsou v podstatě převzata z knihy Garetha A. Jonese a J. Mary Jonesové: Information and Coding (Jones a Jones, 2000, Theorem 3.7-3.10). Tvrzení 2.1.3 jsem dokázal sám, ale předpokládám, že se jedná o standartní tvrzení.
  Část zabývající se přibližnou asymptotikou a ideálním kódováním jsem přidal pro usnadnění značení a nebyla převzata odnikud. Tvrzení 2.2.1, 2.2.2 a 2.2.3 jsou standartní tvrzení teorie pravděpodobnosti a jsou v podobných variantách nalezitelné ve skriptech docenta Hlubinky (Hlubinka, 2018) a knize Olava Kallenberga (Kallenberg, 2002). Ke tvrzením 2.2.4, 2.2.5 jsem nenašel citaci, ale byla probrána na cvičení z předmětu Pravděpodobnost a statistika vedeným doktorkou Šárkou Hudecovou (přednáška pana docenta Hlubinky).
  Většina části zaměřené na typické množiny a SVAR byla převzata z Thomas M. Coverovi a Joy A. Thomasovy knihy: Elements of Information Theory (Cover a Thomas, 2006), kapitoly 3. Jmenovitě se v mé práci jedná o tvrzení 2.3.1, 2.3.2, 2.3.3. Lemma 2.3.4 je poupravené tvrzení Theorem 3.3.1 z téže knihy (není tam k němu uveden úplný důkaz, ale je tam napsaný jakýsi návod v Problem 3.3.11). Věta 2.3.5 je mnou jak naformulována, tak

dokázána na základě inspirace tímto lemmatem. V Coverově a Thomasově knize jsou některé technické detaily ponechány představivosti. Abych kódování C (u nich diskuze na začátku sekce 3.2) zbavil závislosti na ϵ, tak jsem zavedl pojem semioptimální hodnoty chyby π_n a dokázal tvrzení s ním spojená. Cover také explicitně nemluví o párovací funkci, kterou jsem použil v definici 2.3.5 a nezabývá se nastáním kolize v rámci typických (či netypických zpráv). K tomu jsem v dané knize nenašel odůvodnění. Tvrzení 2.3.10, 2.3.11 jsou má.

- **Kolmogorovovská složitost motivace** (kapitola 3.1.) Tato kapitola je primárně historická a motivační. Popisuji jakou potřebu plnila ve své době definice Kolmogorovovské složitosti. Čerpal jsem zejména ze dvou zdrojů. Jednalo se o knihu Ming Liho a Paula Vitányiho: An Introduction to Kolmogorov Complexity and Its Applications (Li a Vitányi, 2008) str. 49-58 a ze článku Christophra P. Porterova: Kolmogorov on the Role of Randomness in Probability Theory (Porter, 2014). Na základě náplně kapitoly mi přišlo vhodné také popsat filosofický spor o frekventivistické vs bayesovské interpretaci pravděpodobnosti. Tato část textu byla z velké části inspirována článkem De Finetti Was Right: Probability Does Not Exist (Nau, 2001) od Roberta F. Nau.

- **Turingovy stroje** (kapitoly 3.2.1-3.2.3) V této kapitole podávám úvod to Turingových strojů. Hlavní historickou inspirací, co se týče klimatu v tehdejší matematické komunitě ve spojení s Turingovými stroji a k Turingovým tezím mi byl článek Orona Shagrira Gödel on Turing on Computability (Shagrir, 2007). Na stranách 27-36 v knize Ming Liho a Paula Vitányiho: An Introduction to Kolmogorov Complexity (Li a Vitányi, 2008) je v kostce shrnuto odkud se bere definice Turingova stroje a základní fakta (bez detailů), která jsou známá. V knize Micheala Sipsera (Sipser, 2012) v kapitolách 3.1 a 3.2 je poskytnuta skvělá intuice, co se týče výpočtu Turingových strojů. Na základě intuice z této knihy jsem formuloval v podstatě všechny definice z kapitoly 3.2.1. Až na definici 3.2.1 zde ovšem nebyly formálně sepsány. To bylo mým úkolem. Abych sloučil Turingovy stroje, které počítají stejně až na přejmenování prvků jsem formuloval definici 3.2.7. Ji a všechny tvrzení s ní spojené jsem formuloval a dokazoval sám. Všechny konstrukce strojů a odůvodnění koncepce algoritmického popisu práce stroje (tzn. celá kapitola 3.2.2) v kapitole 3 (ať už se jedná o příklady 3.2.1-3.2.2 či lemmata 3.2.3-3.2.9) jsem formuloval a dokazoval také já. Jedinou inspirací ke konstrukci strojů byl článek Jasona Teutsche: Universal Turing Machine o konstrukci univerzálního stroje (Teutsch). Z něj jsem čerpal při definici Turingova čísla a při konstrukci univerzálního stroje. V daných konstrukcích (zejména univerzálního stroje) jsem z tohoto článku převzal myšlenku. Konceptuálně a technicky se od něj ovšem kapitoly 3.2.2 a 3.2.3 v podstatě ve všem odlišují.

- **Řešitelná zobrazení a Problém zastavení** (kapitola 3.2.4) V této kapitole se zabývám zejména problémem rozhodnutí. V podstatě celá kapitola směřuje ke konstrukci logiky, kterou jsem dohledal ve článku Gurama Bezhanishviliho a Lawrence S. Mosse: Undecidability of First-Order Logic (Bezhanishvili a Moss, 2019). Byla v ní po částech dána skoro celá konstrukce z definice 3.2.14. Tu jsem upravil, aby vyhovovala mé definici Turingova stroje a doplnil jsem body, které byly ponechány jako cvičení. Zbytek práce včetně toho, jaké se používá ztotožnění Turingových strojů s binárními řetězci s jeho odůvodněním a v podstatě celý formální důkaz věty 3.2.24 byl ponechán představivosti. Z tohoto článku také čerpám velkou část historického kontextu. Zbytek historie o Hilbertově programu jsem našel v prvních třech kapitolách článku Hilbert's Program Then and Now od Richarda Zacha (Zach, 2006) a včetně skvělé intuice o formální logice také v knize Roberta S. Wolfa: A Tour through Mathematical Logic (Wolf, 2005)(kapitol 1.,3.). Výtah z historie teorie modelů včetně příkladu o formálním polynomu/polynomiálním zobrazení mám z recenze Jeremyho Avigada na článek The Birth of Model Theory: Lowenheim's Theorem in the Frame of the Theory of Relatives od Calexto Badesy (Avigad). Všechny formální definice a tvrzení mám nastudovány z knihy Vítězslava Švejdara LOGIKA neúplnost, složitost a nutnost, z kapitol 3.1 a 3.2 (Švejdar, 2002). Problém zastavení je opět možné bez formálních detailů dohledat v knize Michaela Sipsera (Sipser, 2012, Theorem 4.11).

- **Definice a poznatky** (kapitola 3.3) Všechna tvrzení v kapitole 3.3 jsou standartní tvrzení literatury. jejich myšlenky jsou dohledatelné například ve článku Bruno Duranda a Alexandera Zvonkina Kolmogorov Complexity (Durand a Zvonkin, 2007), přičemž k nim jsou doplněny drobné formální detaily. V případě věty 3.27. se jedná o značné formální detaily (včetně potřebné konstrukce stroje).

- **Vztah Kolmogorovovy a Shannonovy teorie**(kapitoly 4.1 a 4.2) Celá první část kapitoly 4.1 směřuje k důkazu Zvonkinovy věty. Ten je naznačen (pro střední hodnotu) ve tvrzení 14.3.1 v knize Thomas M. Coverovi a Joy A. Thomasovy knihy: Elements of Information Theory (Cover a Thomas, 2006). Předělal jsem ji na konvergenci v pravděpodobnosti a všechny formální detaily (tzn. tvrzení 4.1.2-4.1.8) jsem doplnil. Problém použití/odkázání jsem našel ve článku Kolmogorov Complexity and Information Content, Fouada B. Chedida (Chedid, 2017). Na základě něho jsem pro ilustraci vytvořil příklady 4.1.3 a 4.1.4. Hammerova věta v kapitole 4.2 se bez řady formálních detailů vyskytuje ve článku Inequalities for Shannon Entropy and Kolmogorov Complexity od Daniela Hammera, Andreie Romaschenka, Alexandera Shena a Nikolaie Vereshchagina (Hammer a kol., 2000)(jedná s o Theorem 1).

Značení

- **množiny**: Písmeny $\mathbb{N}$ míníme množinu přirozených čísel, $\mathbb{N}_0$ množinu přirozených čísel s nulou, $\mathbb{R}$ množinu reálných čísel, $\mathbb{R}_{>0}$ množinu kladných reálných čísel. Symbol $\subseteq$ značí podmnožinu a $\subsetneq$ vlastní podmnožinu. Pro množiny A,B je $A \times B$ jejich kartézským součinem a $A \setminus B$ jejich množinovým rozdílem.

- **řetězce**: Nechť $\Sigma = \{a_1,a_2,...,a_v\}$ je libovolná konečná množina a $n \in \mathbb{N}$, potom
$$\Sigma^n = \underbrace{\Sigma \times \Sigma \times ... \times \Sigma}_{n}.$$
Dále definuji $\Sigma^+ = \bigcup\limits_{n=1}^{\infty} \Sigma^n$.
Znakem $\emptyset$ budeme myslet prázdný řetězec.
Pro něj definuji $\Sigma^0 = \{\emptyset\}$ a $\Sigma^* = \Sigma^+ \cup \{\emptyset\}$.
Ať $a \in \Sigma^n$ pro nějaké $n \in \mathbb{N}_0$, potom $\ell(a) := n$.
Zápise $\#\ a_i$ v $a \in \Sigma^*$ myslíme počet znaků a_i v řetězci a.
Napíšeme-li pro $a,b \in \Sigma^*$ výraz $a \circ b$, tak tím myslíme a konkatenované s b (tzn. napsaný řetězec a a za ním b).
Někdy, když zapíšeme řetězec tím, že vyjmenujeme jeho znaky, tak jej ohraničíme od zbytku textu uvozovkami, např. '534△'.

- **funkce a operace**: Máme-li nějakou funkci f, potom $Im(f)$ značí její obraz a $D(f)$ její definiční obor. Výrazem $\lceil a \rceil$ pro $a \in \mathbb{R}$ rozumíme horní celou část čísla a a výrazem $\lfloor a \rfloor$ spodní celou část čísla a.

2. Shannonova informace

2.1 Definice a intuice

Roku 1948 byl na světlo světa vypuštěn článek mladého elektroinženýra Clauda E. Shannona, Mathematical Theory of Communication (Shannon, 1948). Ten od základu změnil to, jak je mezioborově chápan pojem informace. Centrálním tématem této práce byla komunikace zpráv po informačním kanále. Přičemž hlavním problémem bylo, jak na jednom místě vybrat nějakou zprávu, zakódovat ji a na nějakém jiném místě ji zpětně zrekonstruovat tak, aby nedošlo ke ztrátě informace. Při tom Shannon narazil na to, jak množství informace ve zprávě měřit. Jeho myšlenka byla až překvapivě jednoduchá.
Pro její ilustraci si představme následující modelovou situaci:

Příklad 2.1.1. *Alice a Bob hrají hru s kostkami. Alice v každém kole hodí kostkou, poskytne Bobovi trochu informace o tom, jaké číslo na ní padlo a vyzkouší, jestli jej Bob bude schopen uhodnout. Tedy Alice hodí kostkou poprvé a následně Bobovi napoví, že na dané kostce padlo nějaké číslo v rozmezí mezi 1 až 6. Bob si po chvíli váhání tipne a zvolí špatně. Inu Alice se proto rozhodne, že mu pomůže o něco více. Hodí kostkou podruhé a nyní mu odhalí, že na kostce padlo číslo sudé. Bobovi se bohužel tip opět nepodaří. Alici se v tu ránu Boba zželí, proto hodí kostkou naposledy a řekne Bobovi, že na ní padla 5. Po kteréžto radě Bob s vypětím všech sil zvolí správnou variantu.*

Je intuitivně zřejmé, že Alice nejméně informace odhalila s první nápovědou, v níž Bobovi o nastálé situaci nesdělila vůbec nic. Potom se druhou a nejvíce se třetí. Když se nad tím zamyslíme, tak ovšem není zcela jasné proč. Například situace, kdy na kostce padne sudé číslo a ta při níž číslo 5 nemohou nastat obě zároveň. Tedy nelze říct, že třetí nápověda ukládá více informace než druhá, protože by druhá šla vydedukovat ze třetí. Nehledě na to, že v našem příkladu se jedná o různé hody.
V podstatě jediným precedentem Shannonovy teorie byla Hartleyho kombinační teorie (Hartley, 1928). Hartley se k daném problému postavil následovně: Zatímco číslo pět může na kostce padnout jen jedním způsobem, tak sudé číslo může padnout způsoby třemi. Čím méně kombinacemi tedy dostanu nějaký hod, tím více informace je spojeno s jeho komunikací.
Tento přístup se zdá být pro tuto modelovou situaci uspokojiví. Některé věci z něj ovšem nejsou zcela zřejmé. Jak bychom kupříkladu porovnávali informaci spojenou se situacemi, kdy jednotlivé kombinace nejsou takto zcela jasně určeny, některé z nich mohou nastávat častěji než jiné, či hůř je jich nekonečně mnoho. Shannon se s touto výzvou vypořádal tak, že využil Kolmogorovovy teorie pravděpodobnosti. Ta vznikla pár let po zveřejnění zmiňovaného Hartleyho článku. Na jejím základě můžeme říct, že protože náhodný jev označující padnutí sudého čísla je více pravděpodobný, než náhodný jev daný padnutím 5, tak nese méně informace. Míra informace je tedy funkce v pravděpodobnosti.
Situace, kterou jsme uvedli výše dozajisté na čtenáře působí zcela uměle. Uvědomme si ovšem, že daný fakt všichni chápeme intuitivně z vlastní běžné zkušenosti. Zdá se být poměrně jasné, že zdaleka ne všechna slova či fráze v každodenní komunikaci nesou stejné množství informace. Tento fakt souvisí s tím, že jak s

danou zprávu, tak s jazykem, ve kterém je formulovaná, se pojí nějaká apriorní očekávání. Tato řekněme struktura nevyhnutelně vede k jisté míře redundance. Vezměme si příklad formálního dopisu. Nikoho z nás by patrně na jeho začátku neudivila slovní spojení jako „Dobrý den“, či „Vážený pane/paní“. Jelikož se dané fráze v tomto kontextu používají takto frekventovaně, tak jsme je v podstatě schopni uhodnout ještě před tím, než danou zprávu vůbec otevřeme. Z hlediska informace jsou tyto fráze v příslušné zprávě v podstatě nadbytečné. Při čtení bychom je z tohoto důvodu dost možná pouze přelétli pohledem. Informace se koncentruje na místech, která neočekáváme. Čím více je tedy nějaká část textu nepředvídatelná tím více informace nese. Jinými slovy množství informace spojené s nějakou částí textu je nepřímo úměrné pravděpodobnosti na její výskyt.
O tom, že proběhla komunikace určitého slova, věty nebo celé zprávy, se dá uvažovat jako o náhodné události. Není těžké si rozmyslet, že v tomto obecném případě dojdeme k analogické intuici. Neboli čím pravděpodobnější je nějaká událost, tím více informace je spojeno s tím, že nastala. Poměrně dobrou volbou na funkci měřící informaci by tedy bylo zobrazení $\frac{1}{P(A)}$, kde $P(A)$ je ona pravděpodobnost jevu A. My bychom ovšem pro nezávislé jevy A,B rádi zachovali aditivitu, tzn. $I(P(A \cap B)) = I(P(A) \cdot P(B)) = I(P(A)) + I(P(B))$. Shannon proto navrhl brát funkci $I(P(A)) = \log_2(\frac{1}{P(A)})$. Slovy Shannona, však hlavní argument pro tuto definici tkví v jejích důsledcích.
Je poměrně zajímavé, že když vypíšeme pár vlastností, které bychom od této funkce očekávali, tak ji určíme v podstatě jednoznačně.
O kterých vlastnostech mluvíme? 1. V první řadě jako funkce v pravděpodobnosti by její definiční obor měl být roven (0,1]. Někdo by mohl namítnout, že pravděpodobnost může být i nulová, ale z očividných důvodů nemá pro jev, který má pravděpodobnost nula smysl kvantifikovat, jak moc informace je spojeno s tím, že nastal. 2. Míra pravděpodobnosti by měla být jistě nezáporná. 3. Poměrně rozumným požadavkem je, aby s malou změnou pravděpodobnosti byla spojena malá změna v míře informace. Tím se v matematice obvykle myslí spojitost. 4. Pro dva nezávislé jevy bychom měli dostat stejně informace, když se dozvíme, že nastaly oba jevy, jako když se dozvíme, že nastal každý z nich zvlášť. 5. Zdá se navíc nepravděpodobné, že bychom si při měření informace vystačili s konstantní funkcí. Máme tedy následující vlastnosti:

1. $D(I) = (0,1]$
2. $I(p) \geq 0, \forall p \in D(I)$
3. I je spojitá funkce (na (0,1])
4. $I(p \cdot q) = I(p) + I(q), \forall p,q \in D(I)$
5. I není konstantní

Nyní tedy přistupme k důkazu (slabé) jednoznačnosti funkce I.

Lemma 2.1.1. *Nechť funkce I splňuje vlastnosti 1.-5., potom pro každé $p \in (0,1]$ a $r \in \mathbb{R}$, $r > 0$ platí*

$$I(p^r) = rI(p)$$

Důkaz. Nejprve si uvědomme, že ze 4.: $I(p^n) = I(p \cdot p^{n-1}) = I(p) + I(p^{n-1})$ pro každé $n \in \mathbb{N}$. Tedy induktivně pro libovolné $n \in \mathbb{N}$ platí $I(p^n) = nI(p)$. Není těžké tento vztah rozšířit na všechna racionální čísla, neboť pro všechna n přirozená odsud plyne $I(p) = I((p^{\frac{1}{n}})^n) = nI(p^{\frac{1}{n}})$, tedy $\frac{1}{n}I(p) = I(p^{\frac{1}{n}})$. Zbytek práce za nás udělá vlastnost 3. Mějme nějaké $r \in \mathbb{R}$. Jelikož $r = \frac{nr}{n} \leq \frac{\lceil nr \rceil}{n} \leq \frac{nr+1}{n} = r + \frac{1}{n}$, tak $\frac{\lceil nr \rceil}{n}$ konverguje z věty o dvou strážnících k r. Ze spojitosti I a obecné mocniny tedy:

$$\begin{aligned} rI(p) &= \lim_{n\to\infty} \frac{\lceil nr \rceil}{n} I(p) \\ &= \lim_{n\to\infty} I(p^{\frac{\lceil nr \rceil}{n}}) \\ &= I(p^{\lim_{n\to\infty} \frac{\lceil nr \rceil}{n}}) \\ &= I(p^r) \end{aligned}$$

□

Věta 2.1.2. *Jestliže funkce I splňuje vlastnosti 1.-5., pak existuje $b > 1$, že*

$$-\log_b(x) = I(x)$$

pro každé $x \in (0,1]$. Na druhou stranu, každá funkce tvaru $-\log_b$ splňuje vlastnosti 1.-5.

Důkaz. Vlastnosti logaritmu jdou dohledat v každé učebnici funkcionální analýzy. Zaměřme se tedy na existenci $b > 0$, že $-\log_b = I$. I je z vlastnosti 5. nekonstantní, tedy existuje $a \in (0,1]$, že $I(a) \neq 0$. Proto z lemmatu 2.1.1

$$0 \neq I(a) = I(e^{\ln(a)}) = I((e^{-1})^{-\ln(a)}) = -\ln(a)I(e^{-1})$$

odsud každopádně $I(e^{-1}) \neq 0$. Z vlastnosti 2. pak $I(e^{-1}) > 0$. Z asymptotiky exponenciály tedy pro $b := e^{\frac{1}{I(e^{-1})}}$ platí $b > 1$. Pro libovolné $x \in (0,1]$ odsud konečně

$$\begin{aligned} I(x) &= -\ln(x)I(e^{-1}) \\ &= -\ln(x)\frac{1}{\ln(e^{\frac{1}{I(e^{-1})}})} \\ &= -\ln(x)\frac{1}{\ln(b)} \\ &= -\log_b(x) \end{aligned}$$

□

Z vlastností 1.-5. tedy vzešla třída funkcí $\{-\log_b, b > 1\}$. Základ logaritmu v tomto případě nehraje nějakou větší filosofickou roli. Jeho volba nese v podstatě charakter výběru jednotek. Je dobré si totiž uvědomit, že dané funkce se liší jenom o přenásobení konstantou. Pro reálná čísla $a,b > 1, p \in (0,1]$ totiž $-\log_b(p) = -\frac{\ln(p)}{\ln(.b)} = -\frac{\ln(a)}{\ln(b)}\frac{\ln(p)}{\ln(a)} = -\log_b(a)\log_a(p)$ V našem případě vybereme základ $b = 2$, jedná se spíše o praktičnost.

Definice 2.1.1. *(Shannonova informace) Nechť A je náhodný jev. Potom množstvím informace jevu A rozumíme číslo*

$$I(A) := \log_2\left(\frac{1}{P(A)}\right)$$

Dodnes se nad obecnou definicí informace vedou vášnivé (filosofické) debaty bez nějakého většího konsenzu. Existuje však jeden motiv, který se vyskytuje v podstatě univerzálně. Informace je něčím, co nám umožňuje vybrat konkrétní možnost z daných alternativ. To reflektuje i jednotka Shannonovy informace. Informace se měří v takzvaných bitech. S tímto pojmem přišel sám Shannon a jedná se o zkratku ze sousloví binary digit, neboli binární číslice. Často se používá jako označení pro prvek množiny $\{0,1\}$. Jeho původní význam je však od obrazu binární veličiny o něco odlišný.
Bit je množství informace, které je třeba na rozlišení mezi dvěma stejně pravděpodobnými disjunktními jevy A_1, A_2, tvořícími rozklad pravděpodobnostního prostoru. Vezměme si například hod poctivé mince. Na rozlišení mezi tím, zda dopadla nahoru panna, nebo orel je třeba právě jeden bit. Výsledek takového pokusu se pak dá zakódovat znaky 0/1. Odpovídající číslice však sama o sobě není bitem. K tomu, aby bylo možné mluvit o informaci je třeba nějaký minimální kontext.
Tuto myšlenku lze zcela přirozeně rozšířit na libovolný konečný počet n takových jevů. Na to abychom zvolili jeden prvek z n možných alternativ tvořících disjunktní rozklad pravděpodobnostního prostoru, je třeba $\log_2(n)$ bitů a daný prvek lze zakódovat $\lceil \log_2(n) \rceil$ znaky z $\{0,1\}$. Lepší představu o tomto faktu dává následující lemma.

Lemma 2.1.3. *Buď B nějaká neprázdná konečná množina.*

- *Dále mějme prosté zobrazení $\phi : B \to \{0,1\}^+$, potom existuje $m \in B$, že $\ell(\phi(m)) \geq \lfloor \log_2(|B|) \rfloor$*
- *Navíc existuje prosté zobrazení $\psi : B \to \{0,1\}^{\lceil \log_2(|B|) \rceil}$*

Důkaz.

- Uvědomme si, že množství navzájem různých prvků, které je možné zakódovat nejvýše k znaky z $\{0,1\}$, je právě $\sum\limits_{i=1}^{k} |\{0,1\}^i| = \sum\limits_{i=1}^{k} 2^i = 2(2^k - 1)$. Buď tedy k největší takové, že $|B| > 2(2^k - 1)$. Jinými slovy k je maximální přirozené číslo splňující, že B nelze zakódovat méně než $k+1$ znaky. Odsud tedy existuje $m \in B$, že $\ell(\phi(m)) \geq k+1$ a jelikož navíc k bylo největší takové, pak jistě $|B| \leq 2(2^{k+1} - 1)$. Odsud tedy:

$$\lfloor \log_2(|B|) \rfloor \leq \lfloor \log_2(2^{k+2} - 2) \rfloor = k + 1 \leq \ell(\phi(m))$$

- Platí, že $|B| = 2^{\log_2(|B|)} \leq 2^{\lceil \log_2(|B|) \rceil} = |\{0,1\}^{\lceil \log_2(|B|) \rceil}|$. Jinými slovy počet prvků B je maximálně takový jako počet prvků $|\{0,1\}^{\lceil \log_2(|B|) \rceil}|$. Tedy prosté zobrazení $\psi : B \to \{0,1\}^{\lceil \log_2(|B|) \rceil}$ jistě existuje.

□

Pokusme se nyní formalizovat množství informace uložené ve zprávě. V podstatě každá forma komunikace, ať už probíhá organicky v přírodě (výměna genetické informace mezi buňkami, ptačí zpěv, včelí tance), nebo mezi lidmi (písemný projev, morseovka, kouřové signály), lze rozdělit na nějakou konečnou atomickou strukturu, nad kterou se dá uvažovat jako nad písmeny. Crick a Watson za nalezení takové chemické 4-písmenné struktury v DNA získali Nobelovu cenu. Pro formu komunikace, která jde rozdělit na konečný počet písmen, můžeme definovat množství informace ve zprávě na základě pravděpodobnostního rozdělení na množině všech zpráv dané délky. Formálně je zpráva m délky n instancí nějakého náhodného vektoru M_n. Množství informace spojené se sdělením této zprávy potom není nic jiného než informace jevu $[M_n = m]$, tedy hodnota $I(m) = \log_2(\frac{1}{P(M_n=m)})$. Základním problém je, že v běžných aplikacích nám dané rozdělení není známé a bez toho, že bychom o něm něco věděli nelze aproximovat. Shannon pracoval extenzivně s více modely tohoto rozdělení. Model, který se v dnešní literatuře pro svou jednoduchost uchytil nejvíce pracuje s písmeny, která jsou z nějakého fixního rozdělení emitována nezávisle na sobě. Někdy se říká, že taková zpráva je generována (silně) stacionárním zdrojem. Za těchto podmínek pak P není problém odhadnout například z empirických četností.

Definice 2.1.2. *Nechť náhodné veličiny* $X_1,X_2,... \sim X$ *tvoří náhodný výběr z* X*. Náhodnou zprávou délky* n *potom rozumíme náhodný vektor* $M_n = (X_1,X_2,...,X_n)$*. Libovolný prvek* $m \in Im(M_n)$ *nazýváme zprávou (délky* n*). Množstvím informace zprávy* m *pak rozumíme číslo:*

$$I(m) := I([M_n = m]) = \log_2(\frac{1}{P(M_n = m)})$$

Informace nám dává nástroj, jak ve chvíli, kdy se dozvíme výsledek náhodného pokusu kvantifikovat míru, jakou je daná událost odchýlená od něčeho, co bychom běžně očekávali. Někdy se tato skutečnost shrnuje ve sloganu, informace je míra překvapení. Představme si nyní, že bychom měli daný výsledek uhodnout. Nejlépe bychom určitě udělali, kdybychom zvolili ten nejpravděpodobnější z nich. Jak jistí si ovšem naším výběrem můžeme být? Lze to nějak měřit? Shannon přišel s překvapivě jednoduchou formulí, která právě tento účel splňuje.

Úmluva. *Pro každou náhodnou veličinu* X *budeme z technických důvodů v průběhu textu mlčky předpokládat, že pro libovolné* $a \in X$ *platí* $P(X = a) > 0$*. Nejedná se o nijak omezující předpoklad. Pro naše účely nedává příliš velký smysl o prvcích s nulovou pravděpodobností mluvit.*

Definice 2.1.3. *Entropií konečné náhodné veličiny* Z *rozumíme číslo:*

$$H(Z) = \sum_{a \in Im(Z)} P(Z = a) \log_2(\frac{1}{P(Z = a)})$$

Jinými slovy entropie určuje střední hodnotu informace $\mathbb{E}I(Z)$, kterou obdržíme s odkrytím hodnoty náhodné veličiny Z. Jak už jsme zmínili, tak se na ni dá z určitého pohledu koukat jako na nějakou míru nejistoty. Termín entropie ve skutečnosti pochází z fyziky. Hraje poměrně důležitou roli ve statistické termodynamice. V jistém smyslu zde vyjadřuje míru neuspořádanosti spojenou se stavem nějakého uzavřeného systému. Máme-li makrokonfiguraci daného systému

(hodnota tepla, tlaku apod.), pak lze entropie spojená s tímto stavem vyjádřil jako $k_b \sum_{\alpha\in W} P(\alpha)\ln(P(\alpha))$, kde W jsou všechny mikrokonfigurace (pozice, rychlost a směr pohybu atomů), které dávají danou makrokonfiguraci. Není těžké si všimnout, že tato definice se liší od té Shannonovy jen přenásobením nějakou konstantou. Jedná se o jednu z prvních paralel mezi fyzikou a Shannonovou teorií.
Podívejme se nyní na nějaké základní vlastnosti entropie.

Tvrzení 2.1.4. *Nechť Z je konečná náhodná veličina, potom*

$$H(Z) \geq 0,$$

kde rovnost nastává právě tehdy, když existuje $a \in Im(Z)$, že $P(Z = a) = 1$

Důkaz. Tvrzení je víceméně jasné, neboť $\log_2(1/x) \geq 0$ pro každé $x \in (0,1]$ přičemž rovnost nastává právě tehdy, když $x = 1$. Druhou část věty dostaneme z toho, že pokud má nějaký prvek obrazu $Im(Z)$ pravděpodobnost rovnou 1, tak je v daném obrazu sám. ☐

Úmluva. *Po zbytek této kapitoly zafixujeme nějaký pravděpodobnostní prostor $(\Omega,\mathcal{F},P)$ a na něm konečnou náhodnou veličinu X takovou, že $Im(X) = \{a_1,a_2.,...,a_l\}$, kde $l \geq 2$ a $P(X = a_i) \in (0,1), \forall i = 1,2,...,l$. Z technických důvodů si nepřejeme aby $H(X) \neq 0$, tedy opomíjíme triviální náhodnou veličinu.*

Nyní se pokusíme stanovit maximum entropie. K tomu budeme potřebovat dvě lemmata.

Lemma 2.1.5. *Pro libovolné x kladné platí $\ln(x) \leq x - 1$ kde rovnost platí právě tehdy, když $x = 1$*

Důkaz. Položme $f = x - 1 - \ln(x)$. Snadno zjistíme, že $f'(x) = 1 - x^{-1}$ a $f''(x) = x^{-2}$. Protože $f'(x) = 0$ právě tehdy, když $x = 1$ a $f''(x) \geq 0$ pro každé $x > 0$, tak f nabývá (ostrého) globálního minima $f(1) = 0$ v bodě $x = 1$. ☐

Lemma 2.1.6. *Nechť $\sum_{i=1}^{n} p_i = \sum_{i=1}^{n} q_i = 1$ pro $p_i,q_i > 0$, potom*

$$\sum_{i=1}^{n} p_i \log_2(\frac{q_i}{p_i}) \geq 0$$

Důkaz.

$$\begin{aligned}
\sum_{i=1}^{n} p_i \ln\left(\frac{q_i}{p_i}\right) &\overset{\text{lemma 2.1.5}}{\leq} \sum_{i=1}^{n}(\frac{q_i}{p_i} - 1)p_i \\
&= \sum_{i=1}^{n} q_i - p_i \\
&= \sum_{i=1}^{n} q_i - \sum_{i=1}^{n} p_i \\
&= 0
\end{aligned}$$

kde rovnost nastává právě tehdy, když $q_i = p_i$ pro libovolné $i = 1,2,...,n$. Ale protože $\ln(2) > 0$ tak nerovnost $\sum_{i=1}^{n} p_i \ln(\frac{q_i}{p_i}) = \ln(2) \sum_{i=1}^{n} p_i \log_2(\frac{q_i}{p_i}) \geq 0$ nastává právě tehdy, když $\sum_{i=1}^{n} p_i \log_2(\frac{q_i}{p_i}) \geq 0$ přičemž rovnost nastává právě tehdy, když $q_i = p_i$ pro $i = 1,2,...,n$ a tím jsme hotovi. ☐

Věta 2.1.7. *Nechť Z je konečná náhodná veličina, kde $|Im(Z)| = n$, potom*

$$H(Z) \leq \log_2(|Z|)$$

kde rovnost nastává právě tehdy, když pro každé $a,b \in Im(Z)$ platí vztah $P(Z = a) = P(Z = b) = \frac{1}{n}$

Důkaz. Ať $Z = \{a_1,a_2,...,a_n\}$ a $P(Z = a_i) = p_i$, potom

$$\begin{aligned} H(Z) - \log_2(n) &= \sum_{i=1}^{n} \Big(p_i \log_2(\frac{1}{p_i})\Big) - \log_2(n) \\ &= \sum_{i=1}^{n} \Big(p_i \log_2(\frac{1}{p_i}) - \frac{1}{n} \log_2(n)\Big) \\ &= \sum_{i=1}^{n} p_i \log_2(\frac{1/n}{p_i}) \\ &\overset{\text{lemma 2.1.6}}{\geq} 0 \end{aligned}$$

kde rovnost z lemmatu 2.1.6 nastává právě tehdy, když $\frac{1}{n} = p_i$ pro každé $i = 1,2,...,n$ ☐

Uvědomme si, že tato vlastnost perfektně sedí s naší intuicí o entropii. Jestliže je pravděpodobnost rovnoměrně rozdělená mezi všechny prvky obrazu, potom si nemůžeme být jisti žádným výběrem hodnoty.
Již jsme zmínili, jaký je vztah mezi informací a entropií. Entropie je míra toho, jak si můžeme být jisti, že se nejpravděpodobnější znak objeví na zdroji a informace je míra překvapení spojená s danými znaky ve chvíli, kdy se je dozvíme. Existuje ovšem ještě jeden fundamentální vztah, který jsme nezmínili. Tento vztah souvisí s kompresí. Dle našeho modelu jsou jednotlivá písmena emitována na základě nějakého fixního rozdělení nezávisle na sobě. Proto by se pro zprávy dostatečné délky měl poměr písmen abecedy v této zprávě blížit jejich teoretickým pravděpodobnostem. Jinými slovy by pro abecedu $Im(X) = \{a_1,a_2,....,a_l\}$ měl:

$$\#a_i \text{ ve zprávě } m = (m_1,m_2,...,m_n) \approx nP(X = a_i)$$

Odsud by z nezávislosti mělo platit

$$P(M_n = m) = P(X_1 = m_1,....,X_n = m_n) \approx \prod_{i=1}^{l} P(X_i = a_i)^{nP(X_i=a_i)} =: \pi$$

Ale tím pádem všechny dlouhé zprávy, na které „typicky“ narazíme mají přibližně stejnou pravděpodobnost π. Tedy, kdybychom se náhodně vybranou zprávu (emitovanou tímto zdrojem) snažili identifikovat, tak bychom v podstatě rozhodovali z přibližně $\frac{1}{\pi}$ alternativ, které všchny nastávají s přibližně stejnou pravděpodobností. My už ovšem z úvodu této kapitoly víme, že množství informace, které potřebujeme na rozhodnutí mezi $\frac{1}{\pi}$ stejně pravděpodobnými jevy je $log_2(\pi)$. Tedy pro zprávu $m \in Im(M_n)$ by mělo platit

$$I(m) \approx \log_2(\pi) = \sum_{i=1}^{n} nP(X_i = a_i)log_2(\frac{1}{P(X_i = a_i)}) = nH(X)$$

V následující podkapitole se budeme zabývat asymptotikou Shannonovy informace o něco blíže a tento důsledek dokážeme formálně.

2.2 Asymptotika informace a ideální kódování

V minulé kapitole jsme zavedli pojem Shannonovy informace a uvedli jsme nějaké základní poznatky o tom, odkud se daná definice bere. Naznačili jsme navíc, že pro náhodnou zprávu M_n délky n emitovanou zdrojovou veličinou X by mělo platit, že $I(M_n) \approx nH(X)$, tzn. informace uložená ve zprávě by měla být přibližně rovna délce zprávy krát entropii veličiny, ze které je daná zpráva emitována. Než tento důsledek dokážeme a uvedeme, co ve skutečnosti myslíme symbolem $\approx$, tak je vhodné se zamyslet nad asymptotikou náhodných veličin.
V této práci budeme používat pojem přibližné asymptotiky. V případě reálných čísel tím budeme myslet následující. Máme-li posloupnosti $\varphi, \psi : \mathbb{N} \to \mathbb{R}_{>0}$, tak řekneme, že se φ asymptoticky chová přibližně jako ψ, či že φ je asymptoticky ekvivalentní s ψ, pokud

$$\frac{\varphi(n)}{\psi(n)} \xrightarrow{n\to\infty} 1$$

V takovém případě totiž $\varphi(n) = \psi(n) + o(\varphi(n)) = \psi(n) + o(\psi(n))$. Tedy φ je pro velká n tak velké jako ψ až na chybu, která je zanedbatelná jak vůči velikosti $\varphi(n)$ tak vůči velikosti $\psi(n)$. Tuto skutečnost budeme značit jako $\varphi \approx \psi$. Hodí se ji zmínit ve chvíli, kdy z nějakého důvodu neplatí $|\varphi(n) - \psi(n)| \xrightarrow{n\to\infty} 0$, ale chceme vyjádřit, že se dané posloupnosti přesto chovají podobně. Stejná intuice stojí za relací: φ není výrazně větší než ψ. Ta platí ve chvíli, kdy pro každé $\epsilon > 0$ a $n \in \mathbb{N}$ dostatečně velké $\frac{\psi(n)}{\varphi(n)} \geq 1 - \epsilon$, což budeme značit $\varphi \preceq \psi$.

Příklad 2.2.1. *Ať $\varphi(n) = n$, $\psi(n) = n + \log_2(n)$, potom $\varphi \approx \psi$, neboť*

$$\lim \frac{\varphi(n)}{\psi(n)} = 1 = \lim \frac{\psi(n)}{\varphi(n)}$$

Rádi bychom tuto intuici rozšířili na posloupnosti náhodných veličin. Je ovšem dobré si uvědomit, že popsat asymptotiku v tomto případě je v lecčems složitější, než je tomu v případě reálných čísel. Z definice zde totiž může každému n místo jedné hodnoty odpovídat zobrazení s poměrně složitým obrazem, kde do toho, která hodnota bude nakonec onou veličinou vybrána vstupuje významným způsobem náhoda. Jak tedy vyjádřit, že se posloupnosti náhodných veličin $\{X_n\}_n, \{Y_n\}_n$ chovají asymptoticky stejně? Mohli bychom říct, že dané veličiny k sobě konvergují právě tehdy, když pro každé $\epsilon > 0$ a $n \in \mathbb{N}$ dostatečně velké $P(|X_n - Y_n| \leq \epsilon) = 1$. Pro posloupnosti, které by tento vztah splňovaly, bychom mohli s jistotou usoudit, že jejich pokročilé členy jsou od sebe vzdáleny o zanedbatelný rozdíl (jako v případě konvergence posloupnosti reálných čísel). Tato definice je ovšem pro naše účely příliš striktní. Jeden způsob, jak ji rozšířit, je připustit, že s nějakou minimální pravděpodobností může nastat jakási anomálie, kdy jsou od sebe dané veličiny vzdálené o více jak o ϵ, za předpokladu, že pravděpodobnost nastání takové anomálii je pro pokročilé členy daných posloupností minimální. Tuto intuici formalizuje takzvaná konvergence v pravděpodobnosti.

Definice 2.2.1. *Nechť $\{X_n\}_n$,$\{Y_n\}_n$ jsou posloupnosti náhodných veličin, potom řekneme že X_n konverguje v pravděpodobnosti k Y_n, pokud pro každé $\epsilon > 0$*

$$P(|X_n - Y_n| \leq \epsilon) \xrightarrow{n\to\infty} 1$$

Tento fakt značíme $X_n \xrightarrow{P} Y_n$

Příklad 2.2.2. *Nechť* $X_1,X_2,\ldots$ *a* $Y_1,Y_2,\ldots$ *jsou posloupnosti náhodných veličin, kde pro každé* n *přirozené* $Im(X_n) = \{0,1\} = Im(Y_n)$, X_n *a* Y_n *jsou nezávislé a* $P(X_n = 1) = P(Y_n = 1) = \frac{n-1}{n}$. *Potom pro libovoné* $\epsilon > 0$ *z věty o úplné pravděpodobnosti (Hlubinka, 2018)*

$$\begin{aligned} P(|X_n - Y_n| \leq \epsilon) &\leq P(X_n = 1 \cap Y_n = 1) + P(X_n = 0 \cap Y_n = 0) \\ &= P(X_n = 1)P(Y_n = 1) + P(X_n = 0)P(Y_n = 0) \\ &= \Big(\frac{n-1}{n}\Big)^2 + \Big(\frac{1}{n}\Big)^2 \xrightarrow{n\to\infty} 1 \end{aligned}$$

Odsud lze nahlédnout, že $X_n \xrightarrow{P} Y_n$ *(resp.* $X_n \xrightarrow{P} 1$*). Jinými slovy pravděpodobnost, že* X_n *a* Y_n *jsou od sebe vzdáleny o méně jak o libovolné kladné číslo* ϵ, *jde k nule:* $P(|X_n - Y_n| > \epsilon) = 1 - P(|X_n - Y_n| \leq \epsilon) \xrightarrow{n\to\infty} 0,$.

V předchozím příkladu si lze všimnout následujícího. Kdyby posloupnosti $\{X_n\}_n$, $\{Y_n\}_n$ splňovaly definici 2.2.1 a my bychom z nich pro jednotlivá n náhodně (dle P) vybírali po jednom prvku, tak bychom shledali, že dané hodnoty jsou k sobě s rostoucím n ve většině případů čím dál blíže a že výkyvy jsou méně a méně časté. Pravděpodobnost nějaké anomálie, tedy toho, že dané hodnoty od sebe budou vzdálené o víc jak o určité $\epsilon > 0$, by totiž měla být z definice pro rostoucí n čím dál blížší nule. Pointou je, že pokud k sobě dvě posloupnoti náhodných veličin konvergují v pravděpodobnosti, pak jsou od sebe tyto veličiny pro velká n skoro jistě vzdálené o zanedbatelný rozdíl (ať zanedbatelným rozdílem myslíme cokoliv). V „reálu" bychom tedy výčet takto tažených hodnot od klasické konvergence nerozeznali.
Pro naši práci jsou poměrně důležité následující čtyři základní vlastnosti konvergence v pravděpodobnosti. U prvních tří neuvádím důkaz. Jedná se o standartní tvrzení.

Věta 2.2.1. *(Sluckého-Cramerova) Nechť* $X,X_1,X_2,\ldots,$ $Y,Y_1,Y_2,\ldots$ *jsou náhodné veličiny, kde* $X_n \xrightarrow{P} X$, $Y_n \xrightarrow{P} Y$, *potom*

$$X_nY_n \xrightarrow{P} XY.$$

Pro každé $a,b \in \mathbb{R}$ *navíc*

$$aX_n + bY_n \xrightarrow{P} aX + bY$$

(Kallenberg, 2002, Corollary 3.5)

Věta 2.2.2. *(Slabý zákon velkých čísel) Buďte* $X_1,X_2,\ldots$ *nezávislé a stejně rozdělené náhodné veličiny s konečnu střední hodnotou* $\mathbb{E}X_1$ *a s konečným rozptylem* $var(X_1)$. *Potom*

$$\frac{1}{n}\sum_{i=1}^{n} X_i \xrightarrow{P} \mathbb{E}X_1$$

(Hlubinka, 2018, Věta 31)

Věta 2.2.3. *(O spojité transformaci) Buďte* $X_1,X_2,\ldots$ *náhodné veličiny a* $a \in \mathbb{R}$ *konstanta, že* $X_n \xrightarrow{P} a$. *Dále mějme funkci* ϕ *spojitou v každém bodě definičního oboru, spňující, že* $Im(X) \subseteq D(\phi)$ *a* $a \in D(\phi)$. *Potom*

$$\phi(X_n) \xrightarrow{P} \phi(a)$$

(Hlubinka, 2018, Konvergence spojité transformace)

Lemma 2.2.4. *Nechť* $X_1,X_2,...,X_n$ *jsou konečné náhodné veličiny tvořící náhodný výběr a* φ *nějaké zobrazení splňující* $Im(X_1) \subseteq D(\varphi)$. *Potom náhodné veličiny* $Y_i := \varphi \circ X_1$, *pro* $i = 1,2,...,n$ *také tvoří náhodný výběr.*

Důkaz. Je poměrně očividné, že $Im(Y_i) = Im(Y_j)$ pro $i,j = 1,2,...,n$ a že dané veličiny jsou stejně rozdělené. Zbývá ukázat nezávislost. Mějme tedy libovolné indexy $i_1 < i_2 < ... < i_k$ z $\{1,2,...,n\}$ a množiny $A_{i_k} \subseteq Im(Y_1)$, potom

$$\begin{aligned} P(\bigcap_k [Y_{i_k} \in A_{i_k}]\ ) &= P(\bigcap_k [X_{i_k} \in \varphi_{-1}(A_{i_k})]\ ) \\ &\overset{\text{nez}}{=} \prod_k P(X_{i_k} \in \varphi_{-1}(A_{i_k})) \\ &= \prod_k P(Y_{i_k} \in A_{i_k}) \end{aligned}$$

□

Důsledek 2.2.5. *Nechť* $X_1,X_2,...$ *jsou náhodné veličiny tvořící náhodný výběr, kde* $Im(X_1) = \{a_1,a_2,...,a_v\}$. *Pro každé* $i = 1,2,...,v$ *a* $n \in \mathbb{N}$ *definujeme náhodnou veličinu* $S_i^{(n)}$ *značící počet* a_i *v náhodném vektoru* $(X_1,X_2,...,X_n)$. *Potom platí*

$$\frac{S_i^{(n)}}{n} \xrightarrow{P} P(X_1 = a_i)$$

Důkaz. Pro každé $i = 1,2,...,v$ uvažujme zobrazení $\chi_i : Im(X_1) \to \{0,1\}$, že $\chi_i(a) = 1$ právě tehdy, když $a_i = a$. Pro něj si uvědomme, že $S_i^{(n)} = \sum_{j=1}^{n} \chi_i \circ X_j$. Protože jsou X_i navíc konečné, tak $\chi \circ X_i$ mají z definice konečnou střední hodnotu a rozptyl. Dle lemmatu 2.2.4 tedy $\chi \circ X_i$ tvoří náhodný výběr. Z věty 2.2.2 odsud konečně

$$\begin{aligned} \frac{1}{n}\sum_{i=1}^{v} S_i^{(n)} &\xrightarrow{P} \mathbb{E}\chi \circ X_1 \\ &= \mathbb{E}\chi(X_1) \\ &= 1 \cdot P(X_1 = a_i) + 0 \cdot P(X_1 \neq a_i) \\ &= P(X_1 = a_i) \end{aligned}$$

□

Konvergence v pravděpodobnosti nám dává mimo jiné nástroj, jak formalizovat přibližné chování náhodných veličin.

Definice 2.2.2. *Nechť* X *je náhodná veličina a* $a \in \mathbb{R}$. *Potom řekneme, že* $X > a$ *pokud* $P[X > a] = 1$. *Analogická definice platí pro ostatní relace* $\leq\ ,\ <\ ,\ \geq\ ,\ =$.

Definice 2.2.3. *Nechť* $\{X_n\}_n, \{Y_n\}_n$ *jsou posloupnosti náhodných veličin, že* $X_n > 0, Y_n > 0$, *potom řekneme, že* X_n *se chová (asymptoticky) přibližně jako* Y_n, *či že* X_n *je asymptoticky ekvivalentní s* Y_n, *pokud*

$$\frac{X_n}{Y_n} \xrightarrow{P} 1$$

nebo-li pokud $\forall \epsilon : P(|\frac{X_n}{Y_n} - 1| < \epsilon) \xrightarrow{n\to\infty} 1$. *Tento fakt značíme* $X_n \approx Y_n$. *Navíc řekneme, že* Y_n *není výrazně větší než* X_n*, tzn.* $Y_n \preceq X_n$*, pokud*

$$P(\frac{X_n}{Y_n} \geq 1 - \epsilon) \xrightarrow{n\to\infty} 1$$

Pro zbytek této práce je vhodné uvést nějaké základní vlastnosti relací $\approx$, $\preceq$. Uděláme tomu tak ve tvrzení 2.2.7. Všimněme si, že v souladu s intuicí se tyto relace v určitém smyslu chovají jako $=, \leq$. Například skutečnost, že se nějaké dvě posloupnosti veličin chovají přibližně stejně by měla jistě být tranzitivní, což zmiňuje hned druhá vlastnost v tomto tvrzení. Ukazuje se dokonce, že $\approx$ je ekvivalencí. Na třídách ekvivalence definovaných $\approx$ navíc $\preceq$ dává částečné uspořádání. Celkový důkaz tohoto tvrzení by byl ovšem zbytečně technický a pro zbytek práce irelevantní. Uvádět jej zde tedy nebudeme. Než se do toho pustíme, tak se pro další postup hodí dokázat následující lemma. 2.2.6

Lemma 2.2.6. *Nechť* A_n,B_n *jsou posloupnosti náhodných jevů a ať* $P(A_n) \longrightarrow 1$ *a* $P(B_n) \longrightarrow 1$. *Potom*

$$P(A_n \cap B_n) \longrightarrow 1 \text{ a } P(A_n \cup B_n) \longrightarrow 1$$

Důkaz. Z věty o inkluzi a exkluzi (Hlubinka, 2018, Věta 2) plyne, že

$$\begin{aligned} P(A_n \cap B_n) &= P(A_n) + P(B_n) - \underbrace{P(A_n \cup B_n)}_{\leq 1} \\ &\geq P(A_n) + P(B_n) - 1 \\ &\longrightarrow 1 + 1 - 1 = 1. \end{aligned}$$

Jenomže z definice pravděpodobnosti platí $P(A_n \cap B_n) \leq 1$, tedy z věty o dvou strážnících $P(A_n \cap B_n) \longrightarrow 1$ a to jsme chtěli. Sjednocení se dokáže analogicky. □

Lemma 2.2.7. *Nechť* $\{X_n\}_n$,$\{Y_n\}_n$, $\{Z_n\}_n$ *jsou posloupnosti kladných náhodných veličin. Jestliže :*

1. $X_n \approx Y_n$*, pak* $Y_n \approx X_n$
2. $X_n \approx Y_n$ *a* $Y_n \approx Z_n$*, potom* $X_n \approx Z_n$
3. $X_n \preceq Y_n$ *a* $Y_n \preceq X_n$*, potom* $X_n \approx Y_n$
4. $Y_n \preceq X_n$ *a* $Z_n \approx Y_n$*, potom* $Z_n \preceq X_n$

Důkaz.

1. Uvědomme si, že platí $\frac{X_n}{Y_n} > 0$ a tedy $()^{-1}$ je spojitá v každém bodě $Im(\frac{X_n}{Y_n})$. Z věty o spojité transformaci 2.2.3 tedy platí

$$\frac{Y_n}{X_n} = \Big(\frac{X_n}{Y_n}\Big)^{-1} \xrightarrow{P} 1^{-1} = 1$$

2. Z předpokladu víme, že $\frac{X_n}{Y_n} \xrightarrow{P} 1$ a $\frac{Y_n}{Z_n} \xrightarrow{P} 1$. Tedy ze Sluckého věty

$$\frac{X_n}{Z_n} = \frac{X_n}{Y_n} \cdot \frac{Y_n}{Z_n} \xrightarrow{P} 1$$

3. Mějme libovolné $\epsilon > 0$. Z předpokladu máme $Y_n \preceq X_n$

$$P\Big(\frac{X_n}{Y_n} \geq 1 - \epsilon\Big) \to 1$$

Položíme-li $\tilde{\epsilon} := \frac{\epsilon}{1+\epsilon} > 0$, pak z předpokladu $X_n \preceq Y_n$ plyne

$$\begin{aligned} P\Big(\frac{X_n}{Y_n} \leq 1 + \epsilon\Big) &= P\Big(\frac{Y_n}{X_n} \geq \frac{1}{1+\epsilon}\Big) \\ &= P\Big(\frac{Y_n}{X_n} \geq 1 - \frac{\epsilon}{1+\epsilon}\Big) \\ &= P\Big(\frac{Y_n}{X_n} \geq 1 - \tilde{\epsilon}\Big) \longrightarrow 1 \end{aligned}$$

Ale tím pádem z lemmatu 2.2.6 dostaneme

$$P\Big(|\frac{X_n}{Y_n} - 1| \leq \epsilon\Big) = P\Big(\frac{X_n}{Y_n} \leq 1 + \epsilon \cap \frac{X_n}{Y_n} \geq 1 - \epsilon\Big) \longrightarrow 1 + 1 - 1 = 1$$

4. Buď $\epsilon > 0$. Z předpokladu víme, že $P(\frac{Y_n}{Z_n} \geq 1 - \epsilon) \geq P(|\frac{Y_n}{Z_n} - 1| \leq \epsilon)$. Tedy triviálně $P(\frac{Y_n}{Z_n} \geq 1 - \epsilon) \to 1$. Dále si uvědomme, že pro $\tilde{\epsilon} := \frac{\epsilon}{2} > 0$ platí $\tilde{\epsilon}(2 - \tilde{\epsilon}) = \epsilon \underbrace{(1 - \frac{\epsilon}{4})}_{\leq 1} \leq \epsilon$, proto $(1 - \tilde{\epsilon})^2 \geq 1 - \epsilon$. Odsud, z předpokladu $Y_n \preceq X_n$ a z lemmatu 2.2.6 pro $\tilde{\epsilon} > 0$ konečně plyne

$$\begin{aligned} P\Big(\frac{X_n}{Z_n} \geq 1 - \epsilon\Big) &= P\Big(\frac{X_n}{Y_n} \cdot \frac{Y_n}{Z_n} \geq 1 - \epsilon\Big) \\ &\geq P\Big(\frac{X_n}{Y_n} \cdot \frac{Y_n}{Z_n} \geq (1 - \tilde{\epsilon})^2\Big) \\ &\geq P\Big((\frac{X_n}{Y_n} \geq 1 - \tilde{\epsilon}) \cap (\frac{Y_n}{Z_n} \geq 1 - \tilde{\epsilon})\Big) \\ &\longrightarrow 1 \end{aligned}$$

□

Nyní již máme dostatečně velkou slovní zásobu na to, abychom formulovali a dokázali větu o asymptotickém chování Shannonovy informace, které jsme odvodili v minulé kapitole. V anglické literatuře se o této vlastnosti někdy hovoří jako o asymptotic equipartition property, nebo-li AEP. V naší práci jej zaměníme za český název slabá vlastnost asymptotické rovnosti.

Tvrzení 2.2.8.

$$\frac{I(M_n)}{n} \xrightarrow{P} H(X)$$

Speciálně $\forall \epsilon > 0 \ \exists n_0 \in \mathbb{N} \ \forall n \in \mathbb{N}, n \geq n_0$

$$P\Big(| - \frac{1}{n}\sum_{i=1}^{n} \log_2(P(X_i)) - H(X)| \leq \epsilon\Big) > 1 - \epsilon$$

Důkaz. Pro libovolnu zprávu $m = (m_1,m_2...,m_n) \in Im(M_n)$ z nezávislosti platí že:

$$\begin{aligned}
\frac{I(m)}{n} &= -\frac{1}{n}\log_2(P(M_n = m)) \\
&= -\frac{1}{n}\log_2(P(X_1 = m_1, X_2 = m_2, ..., X_n = m_n)) \\
&= -\frac{1}{n}\log_2(\prod_{i=1}^{n} P(X_i = m_i)) \\
&= -\frac{1}{n}\sum_{i=1}^{n}\log_2(P(X_i = m_i)) \\
&= \frac{1}{n}\sum_{i=1}^{n} -\log_2(P_{X_i}(m_i))
\end{aligned}$$

Označme tedy $Y_i := -\log_2 \circ P_{X_i} \circ X_i$. Neboť $X_1,X_2,...,X_n$ jsou konečné a tvoří náhodný výběr, pak $Y_1,Y_2,...,Y_n$ jsou konečné a tvoří náhodný výběr z lemmatu 2.2.4. Z konečnosti Y_i navíc plyne $var(Y_i) < \infty$ a $\mathbb{E}Y_i < \infty$. Ze slabého zákona velkých čísel (věta 2.2.2) tedy ihned dostaneme:

$$\begin{aligned}
\frac{I(m)}{n} &= \frac{1}{n}\sum_{i=1}^{n} Y_i \\
&\xrightarrow{P} \mathbb{E}Y_1 \\
&= \sum_{a \in Im(Y_1)} P(Y_1 = a) \cdot a \\
&= \sum_{b \in Im(X_1)} P(X_1 = b))\log_2(P_{X_1}(b)) \\
&= \sum_{b \in Im(X_1)} P(X_1 = b) \cdot \log_2(P(X_1 = b)) \\
&= H(X)
\end{aligned}$$

Tedy ukázali jsme, že $\frac{I(M_n)}{n} \xrightarrow{P} H(X)$. Neboli $\forall \epsilon > 0$ posloupnost $P(|\frac{I(M_n)}{n} - H(X)| \leq \epsilon)$ konverguje k 1. Tedy pro $1 - \epsilon > 0$ $\exists n_0 \in \mathbb{N}\ \forall n \in \mathbb{N}, n \geq n_0$, že

$$P(|\frac{I(M_n)}{n} - H(X)| \leq \epsilon) > 1 - \epsilon$$

a to jsme chtěli. □

Věta 2.2.9. *(Slabá vlastnost asymptotcké rovnosti (SVAR))*

$$I(M_n) \approx nH(X)$$

Důkaz. Z předchozí věty víme, že

$$\frac{I(M_n)}{n} \xrightarrow{P} H(X)$$

Ze Sluckého věty tedy navíc

$$\frac{I(M_n)}{nH(X)} \xrightarrow{P} \frac{1}{H(X)}H(X) = 1$$

A to jsme chtěli. □

V souvislostí s informací nás může napadnout řada logických otázek: Jak moc je informace ve zprávě? Jak se informace chová asymptoticky? Kolik informace připadá v průměru na jeden znak dlouhé zprávy? Uvědomme si, že v naší axiomatice už jsme na ně na všechny odpověděli. Množství informace ve zprávě m je $\log_2(\frac{1}{P(M_n=m)})$. Tato veličina se asymptoticky z důsledku 2.2.9 pro dlouhé zprávy chová přibližně jako $nH(X)$. S pravděpodobností blízkou jedné je z 2.2.8 pro dostatečně dlouhé zprávy průměrné množství informace přicházející na jeden znak přibližně $H(X)$. Všechny tyto body mají poměrně zajímavé důsledky pro kompresi. Než se do vztahu mezi Shannonovou informací a kompresí ovšem pustíme, tak je třeba ustálit nějaké základní názvosloví týkající se kódování.

Definice 2.2.4. *Libovolné prosté zobrazení ψ,*

$$\psi : \bigcup_{n=1}^{\infty} Im(M_n) \to \{0,1\}^+$$

nazveme (binárním) kódováním. O jednotlivých prvcích obrazu budeme někdy mluvit jako o kódových slovech či kódech.

V úvodu jsme mluvili o etymologii slova informace. Připodobnili jsme jej k nějaké ideální identitě zprávy. Identitou v našem případě myslíme obraz zprávy vzhledem k nějakému „rozumnému“ kódování, co se týče komprese. Co ovšem dělá nějaké kódování rozumným či ideálním a jaký je jeho vztah délek kódových slov s Shannonovou informací? Je dobré si uvědomit, že ať zvolíme kódování φ libovolné, tak bude existovat zpráva $m \in Im(M_3)$ a kódování ψ, že $\ell(\psi(m)) < \ell(\varphi(m))$. Tzn. každé kritérium, které vybereme bude nutně minimalizovat délku nějaké zprávy na úkor jiných zpráv. Jaké kritérium tedy dělá kódování objektivně dobré? Ve většině prací se autoři zabývají asymptotickým chováním střední hodnoty délek, to má ovšem svá úskalí (více v kapitole 2.4). My se opět budeme zabývat konvergencí v pravděpodobnosti. Zde přichází na scénu pojem ideálního kódování. Takové kódování splňuje, že jeho délka není výrazně větší jak délka libovolně zvoleného jiného kódování.

Definice 2.2.5. *Řekneme, že kódování ϕ je ideální, pokud pro každé kódování φ platí*

$$\ell(\phi(M_n)) \preceq \ell(\varphi(M_n)).$$

Tato definice jinými slovy říká, že máme-li ideální kódování φ, tak pro každé jiné kódování ϕ jde pravděpodobnost toho, že bude kódové slovo $\phi(m)$ výrazně delší než $\varphi(m)$ pro $m \in Im(M_n)$ (zvolené na základě P), s rostoucí délkou zpráv k nule. Ideální kódování odsud ztělesňuje jakési optimální asymptotické chování, co se týče délky. Jako u každého kódování se sice může stát, že se nějaké jiné kódování bude chovat lépe na jednom, dvou či tisíci zprávách. Ale jak roste délka, tak se dříve nebo později tento rozdíl srovná. Pokud potom pro dlouhé zprávy nějaký rozdíl nastane bývá s délkou dané zprávy zanedbatelný.
Tento předpoklad se zdá být poměrně silný. Někoho by na tomto místě tedy jistě mohlo napadnout, jestli taková definice vůbec někdy může být splněna. Na tuto otázku odpovíme kladně ve tvrzení 2.3.10
Není těžké si uvědomit, že pokud se délka nějakého kódování chová asymptoticky ekvivalentně s délkou ideálního kódování, pak je také ideální a že délky ideálních kódování se chovají asymptoticky ekvivalentně.

Tvrzení 2.2.10. *Jestliže ϕ je ideální kódování a φ je kódování splňující*

$$\ell(\varphi(M_n)) \approx \ell(\phi(M_n)),$$

potom je φ také ideální.

Důkaz. Mějme libovolné kódování ψ. Z předpokladů plyne $\ell(\phi(M_n)) \preceq \ell(\psi(M_n))$ a $\ell(\phi(M_n)) \approx \ell(\varphi(M_n))$. Tedy z lemmatu 2.2.7 dostaneme $\ell(\varphi(M_n)) \preceq \ell(\psi(M_n))$ a to jsme chtěli. □

Tvrzení 2.2.11. *Jestliže ψ_1,ψ_2 jsou ideální kódování, potom*

$$\ell(\psi_1(M_n)) \approx \ell(\psi_2(M_n))$$

Důkaz. Z definice ideálního kódování platí, že $\ell(\psi_1(M_n)) \preceq \ell(\psi_2(M_n))$ a $\ell(\psi_2(M_n)) \preceq \ell(\psi_1(M_n))$. Tedy z lemmatu 2.2.7 ihned $\ell(\psi_1(M_n)) \approx \ell(\psi_2(M_n))$ □

Uvědomme si, že kdybychom na tomto místě věděli o existenci ideálního kódování, pak bychom mohli na základě předchozích dvou lemmat každé ideální kódování charakterizovat následujícím způsobem: ϕ je ideální právě tehdy, když pro každé ideální kódování φ platí $\ell(\varphi(M_n)) \approx \ell(\phi(M_n))$.

2.3 Typické zprávy a C kódování

Tato podkapitola je vyvrcholením naší práce o Shannonově informaci. Jejím cílem je nalezení vztahu Shannonovy informace a délky kódových slov ideálního kódování. Vše, co jsme doposud v tomto kontextu uvedli v podstatě směřuje ke třem tvrzením. Ve tvrzení 2.3.6 ukážeme, že informace uložená ve zprávě je nějakým (přibližným) spodním odhadem na délku kódových slov. Potom nalezneme ideální kódování C, o kterém ve větě 2.3.10 ukážeme, že se délka jeho kódových slov přibližně chová jako Shannonova informace. Ve větě 2.3.11 navíc zjistíme, že tato vlastnost je jak nutnou, tak postačující podmínkou k tomu, aby libovolně zvolené kódování bylo ideální. Určitým způsobem se tedy vrátíme na začátek naší práce, kde jsme pojem informace při líčení filosofického kontextu, ze kterého vzešel, připodobnili k nějaké ideální identitě zprávy. Tím bude náš formalismus kompletní. Z praktického hlediska tím navíc ukážeme, že hranice nastavena Shannonovou informací je pomocí binární komprese přibližně dosažitelná.
Na konci úvodní podkapitoly v této sekci jsme při odvozování SVAR zmínili, že všechny dlouhé zprávy, na které typicky narazíme by měli mít přibližně stejné rozložení písmen abecedy. To by se mělo více méně řídit teoretickými pravděpodobnostmi na jejich výskyt ve zdrojové veličině. O zprávách, které takové kritérium splňují se někdy mluví jako o silně typických. V naší práci se z technických důvodů budeme zabývat zobecněním silně typických zpráv, takzvaně (slabě) typickými zprávami. To jsou takové zprávy, u nichž informace, která v průměru připadne na jeden znak, vychází jako entropie zdrojové veličiny, v našem případě $H(X)$. Není těžké si uvědomit, že slabá typickost je (na silně stacionárním zdroji) opravdu zobecněním silné typickosti. Ze SVAR navíc mají tyto zprávy poměrně zajímavé asymptotické vlastnosti. Ty zjistíme vzápětí.

Definice 2.3.1. *Nechť $\epsilon > 0$. Potom řekneme že $m = (m_1,m_2,...,m_n) \in Im(M_n)$ je ϵ-typickou zprávou délky n, jestliže*

$$H(X) - \epsilon \leq \underbrace{-\frac{1}{n}\sum_{i=1}^{n} \log(P(X_i = m_i))}_{=\frac{1}{n}I(m)} \leq H(X) + \epsilon.$$

Množina všech ϵ-typických zpráv délky n se značí $T_{n,\epsilon}$.

Typickost splývá s naším pojetím něčeho, co je „běžné“. Je poměrně zajímavé, že ne vždy ty zprávy, které jsou nejpravděpodobnější jsou také běžné. Kdyby $Im(X) = \{0,1\}$, $P(0) = 0.9$ a $P(1) = 0.1$, pak by $m = \underbrace{00...0}_{10^6}$ byla sice nejpravděpodobnější zprávou. Běžně bychom ovšem čekali, že přibližně každý desátý prvek m bude roven 1. SVAR tuto naši intuici formalizuje. Navíc říká, že pro dlouhé zprávy bude náš odhad toho, se kterými zprávami se můžeme běžně setkat většinou správný. Následují tři základní charakteristiky typických množin, které dobře demonstrují jejich důležitost.

Tvrzení 2.3.1. *Nechť $m \in T_{n,\epsilon}$, potom*

$$2^{-n(H(X)+\epsilon)} \leq P(M_n = m) \leq 2^{-n(H(X)-\epsilon)}$$

Důkaz. Z definice $T_{n,\epsilon}$ přenásobením nerovnosti číslem $-n$ ihned platí, že

$$-n(H(X)+\epsilon) \leq \log_2(P(M_n = m)) \leq -n(H(X)-\epsilon).$$

Jenomže $2^{log_2(P(M_n=m))} = P(M_n = m)$ a 2^x je rostoucí funkcí. Odsud ihned dostaneme řetězec nerovností z našeho tvrzení. ☐

Tvrzení 2.3.2. *Pro každé $\epsilon > 0$ existuje $n_0 \in \mathbb{N}$, že pro libovolné $n \in \mathbb{N}, n \geq n_0$:*

$$P(M_n \in T_{n,\epsilon}) > 1-\epsilon$$

Důkaz. Z definice absolutní hodnoty a $T_{n,\epsilon}$ je očividné, že

$$T_{n,\epsilon} = \Big\{(m_1,m_2,...,m_n) \in Im(M_n) : |-\frac{1}{n}\sum_{i=1}^{n} \log_2(P(X_i = m_i)) - H(X)| \leq \epsilon\Big\}.$$

Kýžené $n_0 \in \mathbb{N}$ tedy existuje z SVAR viz věta 2.2.9. Tím je důkaz hotový. ☐

Tvrzení 2.3.3. *Pro libovolné $\epsilon > 0$ existuje $n_0 \in \mathbb{N}$, že pro každé $n \in \mathbb{N}, n \geq n_0$*

$$(1-\epsilon)2^{n(H(X)-\epsilon)} \leq |T_{n,\epsilon}| \leq 2^{n(H(X)+\epsilon)}$$

Důkaz.

1. Nejprve ukažme pravou nerovnost. Tvrzení 2.3.1 říká, že pro každé $m \in T_{n,\epsilon}$ a n dostatečně veliké: $2^{-n(H(X)+\epsilon)} \leq P(M_n = m), \forall m \in T_{n,\epsilon}$. Protože to platí pro všechny prvky $T_{n,\epsilon}$, tak v součtu pak:

   $$|T_{n,\epsilon}|2^{-n(H(X)+\epsilon)} \leq P(M_n \in T_{n,\epsilon}) \leq 1$$

   a po přenásobení nerovnosti číslem $2^{n(H(X)+\epsilon)}$ dostaneme: $|T_{n,\epsilon}| \leq 2^{nH(X)+\epsilon}$

2. Zbývá nám levá nerovnost. Horní závora z tvrzení 2.3.1 nám pro dostatečně velké n a $m \in T_{n,\epsilon}$ dává $P(M_n = m) \leq 2^{-n(H(X)-\epsilon)}$. Tedy vezmeme-li tuto nerovnost opět přes všechny prvky $T_{n,\epsilon}$, pak společně s tvrzením 2.3.2 dostáváme:

   $$1-\epsilon \leq P(M_n \in T_{n,\epsilon}) \leq 2^{-n(H(X)-\epsilon)}|T_{n,\epsilon}|$$

   no a odsud konečně $(1-\epsilon)2^{n(H(X)+\epsilon)} \leq |T_{n,\epsilon}|$

☐

Tvrzení 2.3.2 říká, že zvolíme-li ϵ blízké 0 a n dostatečně velké, pak se nějaký prvek z množiny $T_{n,\epsilon}$ na zdroji objeví s pravděpodobností blízkou 1. Kvůli tomu jsou typické zprávy jako pojem nesmírně důležité při zkoumání asymptotického chování kódů. Jedná se o výsek zpráv, jejichž kódová slova reprezentují to, jak se kódování v realitě bude chovat jako celek. Společně tvoří množinu o přibližné velikosti $2^{nH(X)}$ (viz tvrzení 2.3.3). Z logiky věci a tvrzení 2.1.3 by tedy na zakódování náhodně vybrané zprávy délky n mělo být potřeba alespoň $\log_2(|T_{n,\epsilon}|) \approx nH(X)$ bitů. To nám dává jakýsi spodní odhad na binární kompresi. Dokázat tento fakt formálně je již o něco ošemetnější.
K důkazu zmíněného tvrzení nejprve potřebujeme formulovat následující lemma. To v podstatě říká, že máme-li libovolnou posloupnost množin zpráv rostoucích délek, kde každý prvek zmíněné posloupnosti nastává s pravděpodobností větší,

než je nějaká fixní mez, potom jsou pokročilé členy této posloupnosti srovnatelně velké s množinou typických zpráv. Zmíněný fakt jistě není nikterak překvapivý. Typické zprávy koncentrují většinu pravděpodobnosti. Je tedy logické, že nevyhnutelně bude mít každá nezanedbatelně pravděpodobná množina dlouhých zpráv s typickými zprávami nějaký netriviální průnik.

Značení. *Mějme libovolné $n \in \mathbb{N}$, potom pro každé $B \subseteq Im(M_n)$ označme*

$$B^c := Im(M_n) \setminus B$$

Lemma 2.3.4. *Buď $\{n_k\}_k$ rostoucí posloupnost přirozených čísel, že pro každé $k \in \mathbb{N}$ platí $B_{n_k} \subseteq Im(M_{n_k})$. Dále nechť $\delta \in (0{,}1)$, že $\exists k_1 \in \mathbb{N}\ \forall k \in \mathbb{N}, k \geq k_1$:*

$$P(M_{n_k} \in B_{n_k}) > 1 - \delta,$$

potom $\forall \gamma > 0\ \exists k_0 \in \mathbb{N}\ \forall k \in \mathbb{N}, k \geq k_0$:

$$\frac{1}{n_k} log_2(|B_{n_k}|) > H(X) - \gamma$$

Důkaz. Mějme $\gamma > 0$ a položme $\epsilon := \min\{\frac{\gamma}{2}, \frac{1-\delta}{2}\} > 0$. Z lemmat 2.3.2 a 2.3.1 existuje $k_2 \in \mathbb{N}$, že pro každé $k \in \mathbb{N}, k \geq k_2$ platí $P(T_{n_k,\epsilon}) > 1 - \epsilon$ a každé $m \in T_{n_k,\epsilon}$ splňuje $P(M_{n_k} = m) \leq 2^{-n_k(H(X)-\epsilon)}$. Dohromady s předpokladem a deMorganovými pravidly tedy pro každé $k \in \mathbb{N}, k \geq max\{k_1,k_2\}$ platí:

$$\begin{aligned}
1. \qquad P(T_{n_k,\epsilon} \cap B_{n_k}) &= 1 - P((T_{n_k,\epsilon} \cap B_{n_k})^c) \\
&= 1 - P(T^c_{n_k,\epsilon} \cup B^c_{n_k}) \\
&= 1 - P((T_{n_k,\epsilon})^c) - P((B_{n_k})^c) + P((T_{n_k,\epsilon})^c \cap (B_{n_k})^c) \\
&\geq 1 - P((T_{n_k,\epsilon})^c) - P((B_{n_k})^c) \\
&\geq 1 - \epsilon - \delta \\
2. \qquad P(T_{n_k,\epsilon} \cap B_{n_k}) &= \sum_{m \in B_{n_k} \cap T_{n_k,\epsilon}} P(M_{n_k} = m) \\
&\leq 2^{-n_k(H(X)-\epsilon)} |B_{n_k}|
\end{aligned}$$

Kombinací 1. a 2. pak $|B_{n_k}| \geq (1 - \epsilon - \delta) 2^{n_k(H(X)-\epsilon)}$. Jelikož navíc $\epsilon \leq \frac{1-\delta}{2}$, tak $1 - \epsilon - \delta \geq 1 - \frac{1-\delta}{2} - \delta = \frac{1-\delta}{2} > 0$. Tuto nerovnost tedy můžeme zlogaritmovat a tak dostaneme

$$\frac{1}{n_k} log_2(|B_{n_k}|) \geq \frac{1}{n_k}(log_2(1 - \epsilon - \delta) + n_k(H(X) - \epsilon)).$$

Očividně platí, že $\frac{log_2(1-\epsilon-\delta)}{k} \xrightarrow{k\to\infty} 0$. Proto existuje $k_3 \in \mathbb{N}$, že pro každé $k \in \mathbb{N}, k \geq k_3 : \frac{log_2(1-\epsilon-\delta)}{k_3} > -\frac{\gamma}{2}$. Tedy protože $n_k \geq k$, tak odsud a z volby ϵ pro $k_0 := max\{k_1,k_2,k_3\}$ a každé $k \in \mathbb{N}, k \geq k_0$ dostaneme:

$$\frac{1}{n_k} log_2(|B_{n_k}|) \geq \frac{log_2(1 - \epsilon - \delta)}{n_k} + H(X) - \epsilon > -\frac{\gamma}{2} + H(X) - \frac{\gamma}{2} = H(X) - \gamma$$

□

Přistupme nyní k důkazu faktu, ve kterém určíme $nH(X)$ jako spodní řádový odhad délky kódových slov. Myšlenka je následující. Kdyby existovalo kódování, které by onu hranici výrazně podlezlo, pak by jistě existovala vysoce pravděpodobná posloupnost množin $B_n \subseteq Im(M_n)$, jejíž délku by toto kódování minimalizovalo (pod $nH(X)$). Ovšem lemma 2.3.4 nám na velikost takové posloupnosti množin dává jistou omezující podmínku. Jak se ukáže, tak ze zmíněné podmínky jsou tyto množiny tak velké, že kdybychom je opravdu chtěli zakódovat výrazně lépe než na délku $nH(X)$, tak by nám na to v některých případech nevyzbyly bity. Takto bychom dostali spor.

Věta 2.3.5. *Mějme libovolné kódování ψ, potom:*

$$nH(X) \preceq \ell(\psi(M_n))$$

Důkaz. Sporem: Nechť existuje $\zeta > 0$ a kódování ψ, že $P(\frac{\ell(\psi(M_n))}{nH(X)} \geq 1 - \zeta)$ nekonverguje k 1. Tedy $\exists \epsilon > 0 \ \forall \alpha \in \mathbb{N} \ \exists n^{(\alpha)} \in \mathbb{N}, n^{(\alpha)} \geq \alpha :$

$$P(\frac{\ell(\psi(M_{n^{(\alpha)}}))}{n^{(\alpha)}H(X)} \geq 1 - \zeta) < 1 - \epsilon$$

Uvědomme si, že $P(\frac{\ell(\psi(M_{n^{(\alpha)}}))}{n^{(\alpha)}H(X)} \geq 1 - \zeta) = 1 - P(\frac{\ell(\psi(M_{n^{(\alpha)}}))}{n^{(\alpha)}H(X)} < 1 - \zeta)$. Položíme-li tedy $\tilde{\epsilon} := 1 - \epsilon$ pak pro libovolné $\alpha \in \mathbb{N}$ platí

$$P(\frac{\ell(\psi(M_{n^{(\alpha)}}))}{n^{(\alpha)}H(X)} < 1 - \zeta) > 1 - \tilde{\epsilon}. \tag{2.1}$$

Uvažujme nyní pro každé $n \in \mathbb{N}$ množinu

$$B_n := \{m \in Im(M_n) : \frac{\ell(\psi(m))}{nH(X)} < 1 - \zeta\}$$

a posloupnost $\{n_i\}_{i=1}^{\infty}$ takovou, že $n_1 := n^{(1)}$ a pro $i \geq 2$ $n_i := n^{(j)}$, kde j je nejmenší takové, že $n^{(j)} > n_{i-1}$. Potom z (2.1)

$$P(B_{n_k}) > 1 - \tilde{\epsilon}, \ \forall k \in \mathbb{N}$$

Mějme tedy $\gamma := \zeta H(X)$. Z lemmatu 2.3.4 existuje $k_1 \in \mathbb{N}$, že pro libovolné $k \in \mathbb{N}, k \geq k_1$:

$$\frac{1}{n_k} \log_2(|B_{n_k}|) > H(X) - \frac{\gamma}{2} \tag{2.2}$$

Uvědomme si navíc, že pro $k_2 := \lceil \frac{3}{\gamma} \rceil \geq \frac{3}{\gamma}$ a pro každé $k \in \mathbb{N}$, $k \geq k_2$ jistě

$$1 < \frac{3}{2} \leq \frac{k\gamma}{2} = k(H(X) - \frac{\gamma}{2}) - k(H(X) - \gamma)$$

a ekvivalentě

$$k(H(X) - \gamma) < k(H(X) - \frac{\gamma}{2}) - 1 \leq \lfloor k(H(X) - \frac{\gamma}{2}) \rfloor \tag{2.3}$$

Ozačme tedy $l := \max\{k_1, k_2\}$. Z definice B_{n_l} je pro každé $m \in B_{n_l}$

$$\ell(\psi(m)) < n_l(1 - \zeta)H(X) \tag{2.4}$$

Protože je navíc $n_l \geq l$, pak dohromady:

$$\begin{aligned}
\lfloor \log_2(|B_{n_l}|) \rfloor &\overset{(2.2)}{>} \lfloor n_l(H(X) - \frac{\gamma}{2}) \rfloor \\
&\overset{(2.3)}{>} n_l(H(X) - \gamma) \\
&= n_l H(X)\Big(1 - \frac{\gamma}{H(X)}\Big) \\
&= n_l H(X)\Big(1 - \frac{\zeta H(X)}{H(X)}\Big) \\
&= n_l H(X)(1 - \zeta) \\
&\overset{(2.4)}{>} \ell(\psi(m))
\end{aligned}$$

Tedy $\ell(\psi(m)) < \lfloor \log_2(|Bn_l|) \rfloor$ pro každé $m \in B_{n_l}$. To je ovšem spor s lemmatem 2.1.3 □

V tuto chvíli již není těžké dokázat první stěžejní bod této podkapitoly, kterým je role Shannonovy informace jakožto dolní závory na binární kompresy.

Důsledek 2.3.6. *Pro každé kódování ψ platí, že*

$$I(M_n) \preceq \ell(\psi(M_n))$$

Důkaz. Z věty 2.2.9 víme, že $I(M_n) \approx nH(X)$ a z věty 2.3.5 $nH(X) \preceq \ell(\psi(M_n))$. Tedy tvrzení 2.2.7 nám dohromady dává $I(M_n) \preceq \ell(\psi(M_n))$

□

Naším úkolem po zbytek této kapitoly je nalézt ideální kódování a popsat chování délek jeho kódových slov. V předchozím textu jsme se snažili zdůraznit jeden důležitý fakt, který prochází jádrem toho, jak Shannon ve své práci při řešení tohoto problému postupoval. Pointou ϵ typických zpráv $T_{n,\epsilon}$ je, že v porovnání s celkovým počtem zpráv dané délky, je jejich počet poměrně malý. Přesto si uzurpují absolutní většinu pravděpodobnosti. Již jsme totiž ukázali, že pravděpodobnost zbývající na ostatní zprávy je rovna ϵ. Tato skutečnost vedla Shannona k následujícímu triku.
Kdybychom se pro dané délky primárně snažili minimalizovat kódování typických zpráv a ostatní zprávy kódovali v podstatě libovolně dlouze, tak by se to při reálné komunikaci projevilo nejvýše s pravděpodobností rovnou ϵ. Hodnota ϵ v tomto případě tedy nese charakter jakési „chyby“. Ukazuje se, že jak roste délka zpráv, tak je možné tuto chybu volit libovolně blízkou nule. Což je skvělé, neboť naším cílem je pro daná n vybrat ϵ co možná nejmenší. Tak totiž minimalizujeme kód co možná nejužší typické množiny $T_{n,\epsilon}$ soustřeďující většinu pravděpodobnosti. Takových $\epsilon > 0$, že $P(M_n \in T_{n,\epsilon}) > 1 - \epsilon$ může ovšem být nespočetně mnoho. Tedy ideálně malá hodnota ϵ pro určitou délku zpráv nemusí existovat. Nám bude stačit pro dané délky n vybrat nějaké takové hodnoty $\pi_n > 0$, které budou splňovat $P(M_n \in T_{n,\pi_n}) > 1 - \pi_n$ a které půjdou pro rostoucí n k nule. Proto dále budeme zkoumat semi-optimální hodnotu chyby π_n a (semi-optimálně) typické zprávy T_n.

Značení. *Označme $E_n = \{\epsilon > 0 : P(M_n \in T_{n,\epsilon}) > 1 - \epsilon\}$*

Tvrzení 2.3.7. *Existuje* $n_0 \in \mathbb{N}$*, že pro každé* $n \in \mathbb{N}, n \geq n_0$ *platí*

$$E_n \neq \emptyset$$

Důkaz. Buď $\epsilon = \frac{1}{2}$, potom z tvrzení 2.3.2 existuje $n_0 \in \mathbb{N}$, že pro libovolné přirozené $n \geq n_0$ platí

$$P(M_n \in T_{n,\frac{1}{2}}) > 1 - \frac{1}{2} = \frac{1}{2},$$

tedy $\frac{1}{2} \in \{\epsilon > 0 : P(M_n \in T_{n,\epsilon}) > 1 - \epsilon\} = E_n$ pro libovolné $n \geq n_0$.

□

Definice 2.3.2. *Uvažujme* $n_0 \in \mathbb{N}$ *z tvrzení 2.3.7. Potom pro každé* $n \in \mathbb{N}, n \geq n_0$ *vyberme nějaké pevné* $\pi_n \in E_n$*, že* $\pi_n \leq \frac{1}{n} + \inf E_n$*. Řekneme, že* $T_n := T_{n,\pi_n}$ *je množinou semi-optimálně typických zpráv délky n. O číslu* π_n *budeme hovořit jako o semi-optimální hodnotě chyby pro zprávy délky n*

Tvrzení 2.3.8. $\lim\limits_{n\to\infty} P(M_n \notin T_n) = \lim\limits_{n\to\infty} \pi_n = 0$

Důkaz. Nejprve si uvědomme, že $\inf E_n \xrightarrow{n\to\infty} 0$. Buď $\epsilon > 0$, potom dle tvrzení 2.3.2 pro $\tilde{\epsilon} := min\{\frac{1}{2},\epsilon\} \in (0,1)$ existuje $n_0 \in \mathbb{N}$, že pro libovolné $n \in \mathbb{N}, n \geq n_0$ platí: $P(M_n \in T_{n,\tilde{\epsilon}}) > 1 - \tilde{\epsilon}$. Tedy

$$-\epsilon < 0 \leq \inf E_n < \tilde{\epsilon} \leq \epsilon$$

To ovšem znamená, že $\inf E_n \xrightarrow{n\to\infty} 0$. A odsud ihned

$$0 \leq \lim_{n\to\infty} \pi_n \leq \lim_{n\to\infty} \inf E_n + \frac{1}{n} = \lim_{n\to\infty} \inf E_n + \lim_{n\to\infty} \frac{1}{n} = 0 + 0 = 0$$

□

Úmluva. *Pro ta* $n \in \mathbb{N}$*, pro která* T_n *není definováno, dodefinujme* $T_n := \emptyset$ *a* $\pi_n := 2$.

Výše jsme zmínili Shannonův trik na nalezení ideálního kódování. Spočíval v tom zvlášť zakódovat typické zprávy a zvlášť zprávy zbylé. Pro tento účel lze využít například lemma 2.1.3, ve kterém jsme pro libovolnou konečnou množinu B nalezli kódování $B \to \{0,1\}^{\lceil \log_2(|B|) \rceil}$. Toto lemma můžeme použít přes všechny délky jednotlivě na typické a na zbylé zprávy, vše slepit dohromady a takto definované zobrazení prohlásit za naše ideální kódování. Spíše techničtějším problémem, na který při tomto počínání ovšem narazíme je, jak zařídit abychom nepřidělili dvěma zprávám stejný kód. Rozdělením na případy si lze uvědomit, že kolize zde může nastat dvěma způsoby. Buď se nějaké typické zprávě a nějaké netypické zprávě přiřadí stejný kód nebo dvě množiny zpráv stejného typu pro různé délky budou mít shodný počet prvků a kolize nastane tímto způsobem. První z případů má poměrně jednoduché řešení. Před typické a před atypické zprávy můžeme vložit nějaký specifický prefix, který od sebe bude dané množiny odlišovat. Například před typické zprávy lze vložit 0 a před atypické zprávy 1. Tímto způsobem jistě nenastane kolize mezi typickými a atypickými zprávami. Ošetřit druhý typ kolize je o něco složitější. Je jasné, že před kód zprávy potřebujeme nějakým způsobem zakódovat její délku. Ale z očividných důvodů ji tam nemůžeme prostě jen vložit, protože tak bychom mohli vytvořit zase jiné kolize. Zde nám přijde vhod nějaká párovací funkce. Tak se v informatice nazývají prostá zobrazení, která slučují dva (či více) libovolné binární řetězce do jednoho.

Definice 2.3.3. *Libovolné prosté zobrazení* $\psi : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*$ *nazveme párovací funkcí.*

Představme si, že máme dva binární řetězce $a = a_1a_2...a_n, b = b_1b_2...b_m$ a chceme je sloučit nějakým unikátním způsobem dohromady. Nejjednodušší cesta, jak to provést, je zdvojit každý znak obou řetězců, na jejich konec vložit 01 a takto upravené řetězce napsat za sebe. Výsledný řetězec by tedy byl tvaru $a_1a_1a_2a_2...a_na_n01b_1b_1b_2b_2...b_mb_m01$. Postfix 01 by zde kódoval jakousi čárku. Kdybychom od sebe totiž chtěli tyto dva řetězce opět oddělit, tak bychom prostě procházeli celý řetězec zleva doprava, vynechávali každý druhý prvek a ve chvíli, kdy bychom narazili na 01, tak bychom věděli, že tam končí první řetězec a začíná druhý. Ještě si uvědomme, že druhý řetězec odsud v podstatě není třeba transformovat. Často je však třeba párovací funkci používat na více jak dva řetězce a takto se dá zařídit, že dané zobrazení bude univerzálně použitelné.
Další typicky používanou možností, jak párovací funkci definovat je transformovat každý z párovaných řetězců c do tvaru $\bar{c} = 1^{\ell(c)}0 \circ c$, tedy za každý znak c vložit před c jednu jedničku a tento prefix zakončit nulou. Takto upravené prvky by se opět napsaly za sebe. Zde je tedy čárka svým způsobem zakódovaná před místem, kde jsou dané dva řetězce slepeny. I přes jejich jednoduchost ani jednu z těchto párovacích funkcí používat nebudeme. Hlavním důvodem je, že nemají příliš příznivé asymptotické vlastnosti. Délka výsledného kódu by nám po řadě vyšla jako $2\ell(a) + 2\ell(b) + 4$ a $2\ell(a) + 2\ell(b) + 2$. Naše párovací funkce bude krapet složitější za to se bude chovat asymptoticky o něco příznivěji.

Definice 2.3.4. *Pro* $a_1 \in \{0,1\}^*$ *označme*

$$< a_1 >:= \overline{(\ell(a_1))_2} \circ a_1 = 1^{\ell((\ell(a_1))_2)}0 \circ (\ell(a_1))_2 \circ a_1$$

Pro obecné n *a* $a_1,a_2,...,a_n \in \{0,1\}^*$, *kde* $n \geq 2$ *pak definuji*

$$< a_1,a_2,...,a_n >:=<< a_1 >, < a_2, ...,a_n >>$$

Není těžké si rozmyslet, že $< \, , \, >$ definuje párovací funkci. Uvědomme si, že „čárka“ mezi řetězci je zde zakódovaná o něco sofistikovaněji, než tomu bylo v předchozích případech. Před každým z řetězců je vložena jejich délka, která jednoznačně určuje, kde daný řetězec končí. Abychom oddělili binární zápis délky od daného řetězce, tak jsme použili druhou z našich naivních párovacích funkcí. Ještě si všimněme, že výsledná délka kódu $< a >$ pro $a \in \{0,1\}^*$, je v tomto případě $\ell(< a >) = \ell(a) + 2\lfloor \log_2(\ell(a)) \rfloor + 3 = \ell(a) + o(\ell(a))$, což je mírné zlepšení od naivních párovacích funkcí.

Příklad 2.3.1.

$$< 1011,01 >= \underbrace{111}_{1^{\ell(\ell(1011))_2}} \; 0 \; \underbrace{100}_{\ell(1011)_2} \; 1011 \; \underbrace{11}_{1^{\ell(\ell(01)_2)}} \; 0 \; \underbrace{10}_{\ell(01)_2} \; 01$$

Definice 2.3.5. *Pro každé* $n \in \mathbb{N}$ *mějme prostá zobrazení*

$$\psi_n : T_n \to \{0,1\}^{\lceil \log(|T_n|) \rceil} \text{ a } \varphi_n : T_n^c \to \{0,1\}^{\lceil \log(|T_n^c|) \rceil}.$$

Potom definuji kódování

$$C(m) := \begin{cases} 1 \circ < (n)_2, \varphi_n(m) >, \text{pro } m \in T_n \\ 0 \circ < (n)_2, \psi_n(m) >, \text{pokud } m \in T_n^c \end{cases}$$

Poznamenejme, že kódování C není v žádném případě myšleno k praktickému použití. Není v něm totiž nějak explicitně dána souvislost mezi zprávami určité délky n a jejich obrazy. Tedy aby mohl počítač tímto způsobem kódovat, tak by si musel pamatovat obrazy všech 2^n prvků. Paměťové nároky pro toto kódování tedy rostou exponenciálně. Pro nás C nese čistě teoretický význam. Symbolizuje nějaké ideální asymptotického chování, kterého lze na statickém zdroji dosáhnout. K důkazu toho, že je C opravdu ideální již zbývá v podstatě jen ukázat, jak se asymptoticky chová délka jeho kódových slov. Protože typické zprávy koncentrují většinu pravděpodobnosti, tak by dle definice C měla jeho délka přibližně vycházet jako $\log_2(2^{nH(X)}) = nH(X)$. To ukážeme hned v následující větě.

Věta 2.3.9. *Platí, že*

$$\ell(C(M_n)) \approx nH(X)$$

Důkaz. Buď $m \in T_n$, potom z tvrzení 2.3.3:

$$\begin{aligned}
1. \qquad \frac{\ell(C(m))}{nH(X)} &= \frac{1}{nH(X)}\Big(o(n) + \log_2(|T_n|)\Big)\\
&\geq \frac{1}{nH(X)}\Big(o(n) + \log_2((1-\pi_n)2^{n(H(X)-\pi_n)})\Big)\\
&= \frac{1}{nH(X)}\Big(o(n) + \log_2(1-\pi_n) + n(H(X)-\pi_n)\Big)\\
&= \frac{o(n)}{nH(X)} - \frac{\pi_n}{H(X)} + 1 \xrightarrow{n\to\infty} 1\\
2. \qquad \frac{\ell(C(m))}{nH(X)} &= \frac{1}{nH(X)}\Big(o(n) + \log_2(|T_n|)\Big)\\
&\leq \frac{1}{nH(X)}\Big(o(n) + \log_2(2^{n(H(X)+\pi_n)})\Big)\\
&= \frac{1}{nH(X)}\Big(o(n) + n(H(X)+\pi_n)\Big)\\
&= \frac{o(n)}{nH(X)} + \frac{\pi_n}{H(X)} + 1 \xrightarrow{n\to\infty} 1
\end{aligned}$$

Kombinací 1. a 2. jsme tedy ukázali, že $\forall \epsilon > 0\ \exists n_0 \in \mathbb{N}\ \forall n \in \mathbb{N}, n \geq n_0\ \forall m \in T_n :$

$$\Big|\frac{\ell(C(m))}{nH(X)} - 1\Big| \leq \epsilon$$

Ovšem $P(M_n \in T_n) \geq 1 - \pi_n$. To znamená, že po dosazení $\epsilon = \pi_n$ máme

$$\begin{aligned}
P\Big(\Big|\frac{\ell(C(m))}{nH(X)} - 1\Big| \leq \pi_n\Big) &\geq P(M_n \in T_n)\\
&\geq 1 - \pi_n \xrightarrow{n\to\infty} 1
\end{aligned}$$

□

Důsledek 2.3.10. *C je ideální kódování.*

Důkaz. Buď ψ libovolné kódování. Z věty 2.3.5 víme, že $nH(X) \preceq \ell(\psi(M_n))$. Jenomže tvrzení 2.3.9 nám dává $\ell(C(M_n)) \approx nH(X)$. Tedy z tvrzení 2.2.7 plyne $\ell(C(M_n)) \preceq \ell(\psi(M_n))$. To jsme chtěli □

Od úvodní kapitoly jsme směřovali ke vztahu informace a ideálního kódování. Tato věta je nyní už víceméně jen lehkým důsledkem již dokázaných tvrzení. Jak se v ní ukáže, tak pojem Shannonovy informace a délek ideálního kódování jsou přibližně jedno a to samé bez ohledu na to, jaké ideální kódování vybereme. Společně s existencí ideálního C kódování z předchozí věty dávají tyto dvě tvrzení celistvý obrázek o vztahu komprese a Shannonovy informace. Završují tak první kapitolu této práce.

Věta 2.3.11. *Libovolné kódování ψ je ideální právě tehdy, když platí*

$$\ell(\psi(M_n)) \approx I(M_n)$$

Důkaz.

- Buď ψ ideální kódování. Víme, že $\ell(C(M_n)) \approx I(M_n)$ (viz 2.3.10). Jenomže ψ a C jsou ideální, tedy z tvrzení 2.2.11 platí $\ell(\psi(M_n)) \approx \ell(C(M_n))$. Z lemmatu 2.2.7 konečně $\ell(\psi(M_n)) \approx I(M_n)$

- Nechť $\ell(\psi(M_n)) \approx I(M_n)$ a ϕ je libovolné kódování. Potom platí $I(M_n) \preceq \ell(\phi(M_n))$ a tedy z lemmatu 2.2.7 jistě $\ell(\psi(M_n)) \preceq \ell(\phi(M_n))$

□

2.4 Diskuze

Literatura zabývající se statistickým pohledem na binární kompresi se vesměs shoduje na tom, že dlouhé zprávy jdou za určitých podmínek zakódovat $nH(X)$ bity a že lépe to nejde. Zdá se ovšem poměrně problematické vystihnout, co to znamená formálně. Obvykle se při tom pracuje se střední hodnotou délek.(Cover a Thomas, 2006, Theorem 5.4.1)
Střední hodnota je bezpochyby důležitý ukazatel. Vzpomeňme si, že jako parametr nám určuje nějakou v jistém smyslu očekávanou délku ($\min_{a \in \mathbb{R}} \mathbb{E}(X-a)^2 = \mathbb{E}(X)$ (Hlubinka, 2018, str 28)). Často se ovšem opomíná, že sama o sobě nepodává žádnou informaci o tom, jak jsou okolo ní jednotlivé hodnoty rozmístěny (v pravděpodobnosti). Jinými slovy, pokud ukážeme, že střední hodnota délky kódových slov je asymptoticky ekvivalentní s $nH(X)$, tak nemáme záruku, že délka kódu zprávy, kterou náhodně vybereme (dle P), bude s nějakou vysokou pravděpodobností poblíž $nH(X)$. Z daného faktu totiž neplyne nic o tom, jak se asymptoticky chová variance délek. Hlavním přínosem této práce v rámci kapitoly o Shannonově informaci je věta 2.3.5. V ní jsme ukázali alternativní pohled na $nH(X)$ jako spodní odhad na binární kompresi v podobě přibližného chování náhodných veličin.
Není těžké si uvědomit, že z této věty poměrně přímočaře plyne, že $nH(X)$ je spodním odhadem i na chování střední hodnoty. Z tohohle pohledu je tedy naše věta 2.3.5 o něco obecnější.

Tvrzení 2.4.1. *Nechť ϕ je kódování a buď $\epsilon > 0$ libovolné. Potom existuje $n_0 \in \mathbb{N}$, že pro každé $n \in \mathbb{N}$, $n \geq n_0$:*

$$\frac{\mathbb{E}\ell(\phi(M_n))}{nH(X)} \geq 1 - \epsilon$$

tzn. $nH(X) \preceq \mathbb{E}\ell(\phi(M_n))$

Důkaz. Buď $\epsilon > 0$ a položme $\tilde{\epsilon} := \min\{\frac{1}{4}, \frac{\epsilon}{4}\}$. Z věty 2.3.5 existuje $n_0 \in \mathbb{N}$, že pro každé $n \in \mathbb{N}, n \geq n_0$:

$$P(\frac{\ell(\phi(M_n))}{nH(X)} \geq 1 - \tilde{\epsilon}) \geq 1 - \tilde{\epsilon}$$

Potom, ale pro $B_n := \{m \in Im(M_n) : \frac{\ell(\phi(m))}{nH(X)} \geq 1 - \tilde{\epsilon}\}$ platí:

$$\begin{aligned}
\sum_{m \in Im(M_n)} \frac{\ell(\phi(m))}{nH(X)} P(M_n = m) &\geq \sum_{m \in B_n} \frac{\ell(\phi(m))}{nH(X)} P(M_n = m) \\
&\geq \sum_{m \in B_n} (1 - \tilde{\epsilon}) P(M_n = m) \\
&= (1 - \tilde{\epsilon}) P(M_n \in B_n) \\
&\geq (1 - \tilde{\epsilon})^2 \\
&= 1 - \tilde{\epsilon}(2 - \tilde{\epsilon})
\end{aligned}$$

Ovšem pokud $\tilde{\epsilon} - \frac{1}{4}$, tak z definice $\tilde{\epsilon}$:

$$\epsilon \geq 1 > \frac{7}{16} = \frac{1}{4}(2 - \frac{1}{4}) = \tilde{\epsilon}(2 - \tilde{\epsilon})$$

Jestliže naopak $\tilde{\epsilon} = \frac{\epsilon}{4}$, pak $\epsilon < 1$, tedy

$$\epsilon \geq \epsilon(\frac{1}{2} - \frac{\epsilon}{16}) = \frac{\epsilon}{4}(2 - \frac{\epsilon}{4}) = \tilde{\epsilon}(2 - \tilde{\epsilon})$$

V každém případě $\epsilon \geq \tilde{\epsilon}(2 - \tilde{\epsilon})$ pro každé $n \in \mathbb{N}$ dostatečně velké. Dohromady tedy máme $\mathbb{E}\frac{\ell(\phi_n(M_n))}{nH(X)} \geq 1 - \epsilon$. A to jsme chtěli. $\square$

Sám Shannnon ukázal něco podobného našim větám 2.3.6, 2.3.9 pro blokové kódy. Ona věta je známá jako Shannon's source coding theorem. (Shannon, 1948) a v literatuře se vyskytuje poměrně často. Ve zkratce, Shannon uvažoval pro každé $n \in \mathbb{N}$ kódy, kde každý z nich zobrazoval nějakou $A_n \subseteq Im(M_n)$ do indexové množiny $\{1,2,...,|A_n|\}$. Takový kód by zahlásil chybu pokaždé, kdy by se mu na vstupu objevila zpráva mimo A_n. Kódování tohoto typu Shannon nazýval (n-tým) blokovým kódem. Ty pak charakterizoval kódovým poměrem (=coding rate) $R_n = \frac{1}{n}\log_2(|A_n|)$ a pravděpodobností chyby $P_e^n = P(M_n \notin A_n)$. Když potom zkoumal dané parametry asymptoticky, tak dokázal, že existuje posloupnost $n-$tých blokových kódů jejichž parametry splňují $R_n \longrightarrow H(X)$ a $P_n^e \longrightarrow 0$ pro $n \to \infty$. Jeho argumentace při tom byla podobná té naší z věty o C kódování 2.3.9. Zajímavější je však druhá část této věty, tzv. converse part of Shannon source coding theorem, která určila, že pro každou takovou posloupnost, pro níž existuje $\zeta > 0$, že $R_n \leq H(X) - \zeta$ pro všechna $n \in \mathbb{N}$, platí $P_n^e \longrightarrow 1$.
Zmíněná věta bývá obvykle interpretována následovně: Máme-li nějaký kód, potom pravděpodobnost na výsek zpráv, ve kterém na jedno písmeno vychází méně (o nějakou fixní mez) než $H(X)$ znaků 0/1 jde s rostoucí délkou zpráv k nule (Wikipedia contributors, 2019b). S tím se ovšem vynořuje řada otázek. Prvně R_n má, z toho, co jsem vytušil, při této intepretaci určovat kolika znaky z 0/1 lze zakódovat index dané zprávy v $\{1,2,...,|A_n|\}$. Proč jde ovšem každý prvek $\{1,2,...,|A_n|\}$ zakódovat nejlépe $\log_2(|A_n|)$ bity? A co to v tomto případě znamená nejlépe (střední hodnota, konvergence v pravděpodobnosti,...)? Jakou v danou chvíli hraje roli pravděpodobnost jednotlivých prvků? Ve standartní literatuře ((Yeung, 2008, str. 104-105), (Shannon, 1948, sekce 13)) se mi na tyto otázky nepodařilo dohledat uspokojivou formální odpověď. Kdybychom však chtěli tuto větu aplikovat na asymptotiku délek kódů, tak je třeba na tyto otázky odpovědět. Mě osobně nenapadá, jak by se v případě zkoumání konvergence délek kódů v pravděpodobnosti dalo vyhnout nějaké analogii kroků, které jsme provedli v důkazu věty 2.3.6.
Nakonec si všimněme, že při důkazu všech tvrzení o asymptotice délek kódů jsme, jinak než ve formě SVAR, vůbec nepoužívali nezávislost náhodných veličin. Ukazuje se, že vlastnost SVAR platí i v obecnějších případech, než je náhodný výběr. Shannon-McMillanova-Breimanova věta například říká, že SVAR platí pro stacionární ergodické procesy. Stacionární ergodicita zcela zjednodušeně znamená, že daná posloupnost má nějaké homogenní vlastnosti týkajících se pravděpodobnosti výskytu jednotlivých podřetězců a že odhad jejích parametrů (výběrový průměr-střední hodnota,..) konverguje k jejich reálné hodnotě. Tímto způsobem se dá zbavit předpokladu nezávislosti.

3. Kolmogorovovská složitost

3.1 Motivace

Pokusme se nyní shrnout kritiku, která se proti Shannonově teorii občas uplatňuje. O tom, že Shannonova teorie neřeší sémantiku jsem mluvil v úvodu. Tato výtka bývá v literatuře pravděpodobně nejčastější. Existuje ovšem i kritika formálního charakteru. Vše se týká vlastností onoho pravděpodobnostního rozdělení. V první řadě, aby člověk mohl o Shannonově informaci něco rozumného tvrdit, tak musí ve většině případů o daném rozdělení něco netriviálního předpokládat. Obvykle se předpokládá, že písmena tvoří náhodný výběr. V obecnější variantě se pracuje se stacionárními ergodickými procesy. V praktických aplikacích se ovšem zdá, že se v obou případech jedná o poměrně silné zjednodušení reality. Stacionarita by znamenala, že pravděpodobnost výskytu každé konečné posloupnosti písmen by byla všude v textu stejná. To se však nezdá být pravdivé. Co se týká aplikovatelnosti samotné teorie však Shannonova definice pravděpodobně největší problém zdědila ze základů axiomatické teorie pravděpodobnosti.

Roku 1933 ruský matematik Andrej N. Kolmogorov vydal průlomový článek Grundbegriffe der Wahrscheinlichkeitsrechnung, díky němuž se teorie pravděpodobnosti stala všeobecně uznávanou matematickou disciplínou. V něm axiomaticky zavedl pravděpodobnost na základě nějakých jednoduchých vlastností (spojených s množinami elementárních jevů), které bychom od takové definice očekávali. Kolmogorovým primárním cílem bylo nalezení axiomů, se kterými by se dobře manipulovalo (Li a Vitányi, 2008, str. 50). Z tohoto důvodu ovšem zmiňovaná teorie už ze své podstaty nemohla odpovědět na některé fundamentální otázky: Vezmeme-li libovolný náhodný jev, jako padnutí rubu na hozené minci, lze zaručit, že pravděpodobnost daného jevu bude existovat? A pokud daná hodnota existuje, jaká je její interpretace? Jak ji v praxi odhadneme? V následujícím textu se v kostce pokusím nastínit dva většinové postoje ke zmíněným problémům. Skvělý detailní popis lze nalézt například v (Nau, 2001)

To, jak lidé dodnes přistupují k interpretaci pravděpodobnosti, se dá víceméně rozdělit na dva proudy: na frekventisty a bayesovce. Tradičně byla pravděpodobnost vnímána z frekventistické perspektivy. Frekventisté berou pravděpodobnost jako danou vlastnost náhodné události. Pravděpodobnost podle nich vyvstává jako relativní frekvence opakování příslušného procesu za stejných či podobných podmínek. Pomineme-li praktičnost frekventivistického pohledu na věc, tedy že žádná událost nemůže nastat dvakrát za zcela stejných podmínek (už vůbec ne nekonečněkrát po sobě) a že definovat podobné podmínky, aby jejich rozdíly výsledek daného pokusu neovlivňovaly je skoro nemožné, tak i přesto dojdeme k poměrně podstatnému problému. V první řadě není nijak zaručeno, že daná limita relativní frekvence úspěchu vůbec existuje. Nemůže se navíc stát, že budeme-li provádět dvě série pokusů paralelně a nezávisle na sobě, že se budou blížit k různým hodnotám? To vše motivovalo De Finettiho a jeho následovníky jako Ramsaye, Savage a Lindleyho, k práci na subjektivní míře pravděpodobnosti. Pojícím motivem jejich teorií bylo, přijetí faktu, že objektivní míra pravděpodobnosti nemusí existovat. Pravděpodobnost v jejich pojetí byla pouze míra toho, jak je jednotlivec ze své zkušenosti ochoten vsadit na nastání nějaké události. Onen člověk se samozřejmě může při svém rozhodování opřít o jakýkoliv frekventistický či logický aparát. Jak k dané hodnotě ovšem nakonec došel není podstatné. Na čem záleží je pouze ona hladina jistoty.

Pro ilustraci si představme, že chceme hodit dvoustrannou mincí. Jaká je pravděpodobnost, hodíme-li jí do vzduchu, že dopadne rubem nahoru? Můžeme přijmout frekventivistický přístup, podívat se na předešlé pokusy a říct, že poměr toho kolikrát nahoru dopadnul rub mince je přibližně stejný jako počet událostí, kdy spadl líc a tedy přiřadit této události pravděpodobnost rovnou jedné polovině. Co když je ovšem něčí předešlá zkušenost diametrálně odlišná a rub mu spadl jenom v poměru $1:3$? Nemůžeme navíc někdo zaujmout klasický přístup a říct si, že mince má dvě strany, zákony fyziky jí umožňují dopadnout stejně dobře oběma těmito stranami nahoru, a k hodnotě $\frac{1}{2}$ dojít takto? Jinými slovy každý člověk může ze zkušenosti dojít k různým hodnotám pravděpodobnosti, ale i k jedné hodnotě může dojít růzými způsoby. Pravděpodobnost je tedy z tohoto pohledu relativní a určuje naše vnitřní přesvědčení o tom, že daná událost nastane. Jenomže tím jsme v podstatě nepochopený termín pravděpodobnost pouze zaměnili za jiný abstraktní termín vnitřní míra přesvědčení. Je vůbec možné takovou míru objektivně měřit? Kdy je tato hodnota $\frac{1}{\pi e^2}$ a kdy zase $\frac{1}{8}$? Jak by taková měřící metoda vůbec probíhala?

Důležité je si tedy uvědomit, že ať už zaujmeme kteroukoliv pozici k interpretaci pravděpodobnosti, tak v reálu nemusí tato hodnota vůbec existovat.

Z některých pramenů jako (Kolmogorov, 1998) lze vytušit, že Kolmogorov viděl řešení zmíněné otázky aplikovatelnosti v nějaké úpravě Von Misesovy práce na frekventistické teorii pravděpodobnosti. Ta byla hlavním precedentem jeho již zmiňovaného článku z roku 1933. Bohužel se potýkala s řadou problémů. Na konferenci v Ženevě roku 1937 Maurice R. Fréchet formuloval seznam formálních nedostatků, které daná definice obsahovala a tím, i přes některé pozdější snahy o její reformulaci, byla v podstatě ponechána v propadlišti dějin. (Lambalgen, 2002)

Von Mises zadefinoval pravděpodobnost jako limitu relativní frekvence úspěchu v posloupnosti, za podmínky, že daná posloupnost splňovala jistý předpoklad náhody (formální detaily například v (Porter, 2014)). Ten zajišťoval, že určitá pravidla pravděpodobnostního kalkulu budou při počítání s těmito relativními frekvencemi zachována. Kromě toho, že existence zmíněné limity se ukázala jako poměrně silný předpoklad, tak hlavním problém se stala právě podmínka náhody. Nalézt definici náhody bez odvolání k existenci pravděpodobnosti se stalo poměrně těžkým oříškem. Sám Kolmogorov byl informační teorií fascinován do určité míry právě proto, že v ní viděl způsob, jak se s tímto problémem popasovat (Porter, 2014).

Asi není třeba zmiňovat, že náhoda je poměrně silně spjata s informací. Čím je nějaká posloupnost náhodnější, tím méně je předvídatelná, o to méně je tím pádem také redundantní a o to více informace tedy nese (Gleick, 2011, str.343). Kolmogorovova snaha proto tkvěla v nalezení míry informace, která by byla schopna měřit informaci individuálních řetězců bez nějakých dodatečných pravděpodobnostních předpokladů. Klíčovou myšlenkou v dané teorii byl popis zprávy. Ilustrujme tuto myšlenku na příkladu:

Představme si, že jsme určitým způsobem bez nějakých další znalostí obdrželi řetězec:

$$000000000000000000000000000000$$

nic dalšího o tomto řetězci nevíme. Jak moc informace jsme tímto dostali? Intuice nám říká, že moc asi ne. Je to jenom nula napsaná třicetkrát po sobě. Na druhou stranu je-li ten řetězec například

$$\underbrace{10}_{2}\underbrace{11}_{3}\underbrace{101}_{5}\underbrace{111}_{7}\underbrace{1011}_{11}\underbrace{1101}_{13}\underbrace{10001}_{17}$$

tak se nám očividně někdo snaží sdělit prvních pár prvočísel. Pointou je, že zatímco první posloupnost jde popsat poměrně snadno: napiš nulu 30-krát za sebe, tak kdybychom se snažili popsat druhou posloupnost, tak bychom museli nejen předat, kolik prvočísel je v této posloupnosti napsáno, ale také popsat co to vůbec prvočíslo je (jak jej otestovat). Složitost nejjednoduššího algoritmu, který takovou posloupnost emituje se dá brát jako míra vnitřní informace onoho řetězce. Kolmogorov tedy informaci definoval jako minimální popis zprávy, ze kterého je zpětně zrekonstruovatelná.

Pro tuto definici samozřejmě musíme nejprve určit, co myslíme algoritmickým popisem. Při tom je třeba abychom postupovali poměrně opatrně. Neformální uchopení slova popis bylo za historii matematiky původem řady matematických paradoxů. Asi nejznámějším je takzvaný Berryho paradox:

s je nejmenší číslo, které nejde popsat méně jak 1000 slovy

Touto větou jsme jednoznačně popsali číslo s. Problém je, že zmíněný popis má méně jak tisíc slov. To nás vede ke sporu. Spor je právě v tom, jak si můžeme vyložit slovo popis. V matematice se tím v podstatě výhradně myslí algoritmický popis. O něm se dozvíme v následující kapitole.

Cílem této kapitoly bylo ukázat potřebu pro alternativní definici informace. Přijde mi vhodné ji tedy zakončit následujícím Kolmogorovovým citátem.

> *„Jaký smysl, ale má například otázka typu, kolik informace je obsaženo v Tolstojově Válce a Míru? Je rozumné uvažovat daný román jako prvek množiny všech „možných románů“, či na této množině vůbec postulovat existenci náhodného rozdělení?“*

(Kolmogorov, 1968, str. 162)

3.2 Turingovy stroje

3.2.1 Základní definice a poznatky

Mnoho matematiků bylo na přelomu devatenáctého a dvacátého století fascinováno lidským uvažováním a otázkou, zda-li je tato činost mechanizovatelná. Z části byla fascinace této frakce matematické komunity jistě motivována tím, že čím dál častěji se vynořovaly otázky, které narážely na to, co je a co není mechanicky dokazatelné. Jedna z nejznámějších takových výzev je známá pod jménem Entschediungsproblem (=problém rozhodnutí). Tento problém byl formulovaný Davidem Hilbertem. V podstatě šlo o to, jestli je možné nalézt postup, který v každé sadě axiomů zjistí, jestli je z nich libovolně zvolené tvrzení dokazatelné, nebo ne. Tento problém a všechny podobné problémy narážely na tu samou překážku. Není možné ukázat, že něco jde (resp. nejde) bez toho, abychom přesně formulovali, co tím myslíme. Neboli bylo potřeba definovat, co to formálně znamená mechanické řešení problému. V dnešní terminologii bychom řekli, že bylo potřeba definovat pojem algoritmus.
Do poloviny třicátých let vznikala řada prací snažících se tento pojem formalizovat. Zmiňme například Churchův lambda kalkulus a Kleeneovi rekurzivní funkce. Některé z těchto formalizací se navíc zdály být poměrně užitečné k rozlousknutí zmíněných problémů. Alonzo Church například ukázal, že v jeho axiomatice algoritmus popsaný zmíněným Entscheidungs problémem neexistuje. Ovšem tato, ani žádná jiná formalizace, nebyla matematickou komunitou nějak široce přijímána. Jedním ze známých skeptiků ohledně lambda kalkulu (jakožto formalizace mechanické procedury) byl například Kurt Gödel ((Zach, 2006, str. 3), (Shagrir, 2007, str. 7)). Status quo přetrvával do doby, než jej prolomil tehdy ještě mladý student na univerzitě v Cambridge, Alan Turing. Ten byl na jednom z kurzů vyučovaných N.W. Newmanem představen Entscheidungs problému. Turinga napadlo velice elegantní a intuitivní řešení, které se na tento problém dívalo z úplně jiného úhlu než jeho předchůdci. Místo toho, aby se zabýval specifickými operacemi a jak je možné je v algoritmu skládat dohromady, tak se podíval, jak při řešení problémů lidé obecně pracují se symboly. Velkou inspirací mu při tom byli takzvaní computers.
Computer bylo zaměstnání, ve kterém jste podle nějaké sady instrukcí vykonávali výpočetní činnost. Slavně například pro NASA zpracovávali a vykreslovali data z testů na větrných tunelech ve výzkumném středisku v Langley (McLennan, 2011). Činili tak pouze za pomocí sady instrukcí definujících danou úlohu, papíru a tužky (případně mechanické kalkulačky). Turing si všimnul, že jejich práce je poměrně pravidelná. Podle Turinga bylo to, jak se computer v každém kroku kalkulace rozhodne popsatelné dvěma veličinami. Na jaké číslo se zrovna kouká a v jakém je computer stavu mysli. Stav mysli reprezentuje například to, jestli si při sčítání člověk drží jedničku, jestli hledá nějaké číslo na papíře, atp. Podle těchto dvou veličin computer udělal nějakou snadnou operaci, napsal si její výsledek na papír a podíval se v sadě instrukcí, co má dělat dál.
Turinga napadlo zkonstruovat jednoduchý matematický model stroje, který by tuto práci napodoboval. Udělal při tom pár omezujících předpokladů. Zaprvé computer podle něj nepotřeboval dvojdimenzionální papír. Papír by totiž mohl být rozstříhaný na jeden dlouhý proužek, na kterém by se člověk mohl pohybovat jenom doleva a doprava. Zadruhé každý computer může v jednu chvíli vnímat

jenom jeden z nějakého konečného počtu symbolů. Je pravda, že například čísel, se kterými teoreticky může pracovat, může být nekonečně mnoho, ale v jednu chvíli je stále schopen přijmout jenom konečný počet číslic. Napíši-li například 999999999 tak si čtenář jistě bude muset toto číslo rozdělit na části skládající se z menšího počtu číslic. Většina lidí si dlouhá čísla, jako telefonní číslo, dělí na trojice. A poslední podmínka je, že podle Turinga člověk může umět v době, kdy daný problém řeší, provádět jenom konečně mnoho operací. Tedy řešená úloha musí být popsatelná v konečně mnoha stavech mysli (to je jediný bod který byl ve své době trochu kontroverzní (Shagrir, 2007)).
Jednou z Turingových tezí bylo, že model, který vytvořil, uměl libovolnou úlohu, která tyto podmínky splňovala, řešit. Teoreticky se tento stroj skládal ze z jedné strany nekonečné pásky, čtecí/zapisovací hlavy a konečné množiny vnitřních stavů Q odpovídajících stavům mysli. Při práci stroj manipuloval s nějakou konečnou množinou vstupních symbolů Σ (reprezentující například čísla). Ta políčka, která stroj ještě nezměnil a ta která vymazal, byla vyplněná speciálním prázdným symbolem ϵ. Symboly si stroj navíc mohl při práci podobně jako člověk všemožně označovat. Přičemž množina všech těchto symbolů dohromady tvořila pracovní abecedu A tohoto stroje. To, jak na sebe jednotlivé kroky navazovaly určovala transformační funkce δ. V každém kroku hlava přepsala prvek, nad kterým byla nastavena, změnila vnitřní stav a posunula se o jedno políčko doleva, nebo doprava. Stroj počítal do té doby, dokud se nedostal do speciálního koncového stavu q_h.

Definice 3.2.1. *(Turingův stroj) Ať*

- A,Σ,Q *jsou neprázdné konečné množiny, že* $\Sigma \subsetneq A$
- $\epsilon \in A \setminus \Sigma$, $q_h \neq q_s \in Q$
- $\delta : (Q \setminus \{q_h\}) \times A \to Q \times A \times \{1, -1\}$,

potom sedmici $T = (A,\Sigma,Q,\delta,q_h,q_s,\epsilon)$ *nazýváme (deterministickým) Turingovým strojem a:*

- Q *konečnou množinou stavů*
- A *pracovní abecedou,* Σ *vstupní abecedou*
- q_h *koncovým stavem,* q_s *počátečním stavem*
- ϵ *prázdným znakem*
- δ *transformační funkcí*

Definice 3.2.2. *(Binární Turingův stroj) Turingův stroj* T *nazveme binární, pokud* $\Sigma = \{0,1\}$.

Determinismus v tomto případě znamená, že to, jak se stroj zachová, je v každém kroku určeno pouze tím, na jaké pozici pásky se nachází hlava stroje tím, jaký je v danou chvíli na dané pozici symbol a v jakém je vnitřním stavu. Této trojici se také někdy říká konfigurace stroje. Poznamenejme ještě, že v literatuře se pracuje i s nedeterministickými stroji.

Definice 3.2.3. *Nechť* $T = (A,\Sigma,Q,\delta,q_h,q_s,\epsilon)$ *je Turingův stroj, potom trojici* $K = (\alpha,\beta,\gamma) \in A^* \times Q \times \mathbb{N}$, *kde* $\alpha = (a_1,...,a_n)$ *a* $i \leq n$ *nazveme konfigurací stroje* T. *Pokud navíc* $\beta = q_h$, *potom řekneme, že* K *je koncovou konfigurací stroje* T.

Stroj počítá následovně. Na začátku je hlava stroje nastavená na první pozici pásky, stroj je v počátečním stavu a na pásce je vstup následovaný nekonečným řetězcem prázdných znaků. Následující konfigurace je určena transformační funkcí δ. Jak bylo zmíněno, tak podle výstupu δ stroj přepíše symbol, změní vnitřní stav a posune hlavu doleva, nebo doprava. Kdyby při posunu doleva ovšem měl sjet z pásky, tak zůstane na místě. Takto počítá, dokud se nedostane do speciálního koncovém stavu. V něm skončí práci. Vše, co pak na pásce zbyde po vynechání znaků mimo vstupní abecedu, se bere jako výstup. Stroj tedy nedělá nic jiného než jakési řetězení konfigurací.

Definice 3.2.4. *Ať* $K_1 = (\alpha_1,\beta_1,\gamma_1)$ *je nějaká konfigurace stroje* T. *Potom pro* $i \geq 2$ *přirozené, kde* $\beta_{i-1} \neq q_h$ *označíme* $(\phi^{(i)},\psi^{(i)},\pi^{(i)}) := \delta(\beta^{(i-1)},a^{(i-1)}_{\gamma_{i-1}})$ *a definujeme* i*-tou konfiguraci* K_i *(stroje* T*) při vstupní konfguraci* K_1 *následovně:*

- *Položíme* $\beta_i := \psi^{(i)}$, $\gamma_i := max\{1,\gamma_{i-1} + \pi^{(i)}\}$,
  $a^{(i)}_j := \begin{cases} a^{(i-1)}_j, \ j \neq \gamma_{i-1} \\ \phi^{(i)}, \ j = \gamma_{i-1} \end{cases}$ *přes všechna* $j \leq n_{i-1}$ *přirozená*
  a pro $n_i := max\{\gamma_i,n_{i-1}\}$ *v případě* $n_i = n_{i-1} + 1$ *dodefinujeme* $a_{n_i} := \epsilon$.
  Celkově potom máme $K_i := (\alpha_i,\beta_i,\gamma_i)$, *kde* $\alpha_i := (a^{(i)}_1...a^{(i)}_{n_i})$

Pokud pro nějakou konfiguraci L *stroje* T *existuje* $i \in \mathbb{N}$, *že* $K_i = L$, *potom řekneme, že konfigurace* K_1 *implikuje či dává (na stroji* T*) konfiguraci* L. *Tento fakt značíme* $K_1 \xrightarrow{T} L$. *O posloupnosti* $K_1,K_2,...,K_n$ *někdy budeme mluvit jako řetězci konfigurací stroje* T, *budeme jej značit* $K_1 \rightarrow K_2 \rightarrow ... \rightarrow K_n$

Definice 3.2.5. *Nechť* $T = (A,\Sigma,Q,\delta,q_h,q_s,\epsilon)$ *je Turingův stroj a mějme vstup* $\alpha = a_1...a_n \in \Sigma^*$ *do* T. *Potom pro* $\hat{\alpha} := \begin{cases} \alpha, \ \alpha \neq \emptyset \\ \epsilon, \ \alpha = \emptyset \end{cases}$ *uvažujme vstupní konfiguraci* $K_s = (\hat{\alpha},q_s,1)$. *Ať navíc* $K_h = (\beta,q_h,i)$, *kde* $\beta = b_1....b_m \in A^*$ *je koncová konfigurce stroje* T *taková, že* $K \xrightarrow{T} K_h$. *Potom definuji*

$$T(\alpha) = T(a_1....a_n) = b_{i_1}...b_{i_l}$$

kde $b_{i_1}....b_{i_l} \in \Sigma^*$ *je řetězec vybraný z* β *splňující, že* $\forall j \in \{1,2,...,m\} \setminus \{i_1,i_2,...,i_l\}$ *platí* $b_j \notin \Sigma$. $T(\alpha)$ *se pak nazývá výstup* T *při vstupu* α. *O* K_h *někdy budeme mluvit jako o koncové konfiguraci (stroje* T*) implikovanou vstupem* α. *Pokud* K_s *žádnou koncovou konfiguraci neimplikuje, potom řekneme, že* T *nedefinuje výstup na vstupu* α, *či že* $T(\alpha)$ *není definovaný.*

Na tomto místě je možná dobré si uvědomit, že zdalekane na všech řetězích musí libovolný Turingův stroj definovat výstup. Ilustrujme to na následujícím příkladě

Příklad 3.2.1. *Mějme stroj* $T= (\{0,1,\epsilon\},\{0,1\},\{q_s,q_h\},\delta,q_h,q_s,\epsilon))$, *kde* δ *je určeno vztahy:*

$$(q_s,1) \rightarrow (q_s,1,1), \quad (q_s,0) \rightarrow (q_s,0,1), \quad (q_s,\epsilon) \rightarrow (q_s,\epsilon,1)$$

Potom T *na libovolném vstupu nikdy neskončí práci.*

Stroj z příkladu 3.2.1 se na libovolném vstupu nikdy nepřepne na jiný než počáteční stav. Odsud nemá šanci se dostat do stavu koncového. Je tedy v nekonečném cyklu. Množina všech vstupů, na kterých tento stroj skončí práci, je jinými slovy prázdná. Analogicky by se dalo říct, že definiční obor tohoto Turingova stroje je prázdný.

Definice 3.2.6. *(definiční obor TS) Nechť* T *je Turingův stroj. Potom množinu* $D(T) = \{a : T(a) \text{ je definovaný}\}$ *nazveme definičním oborem stroje* T.

Příklad 3.2.2. *Následující binární Turingův stroj vrátí vstup nezměněn.*
Položme T= $(\{0,1,\epsilon\},\{0,1\},\{q_s,q_h\},\delta,q_h,q_s,\epsilon)$, *kde* δ *je určeno vztahy:*
$(q_s,1) \rightarrow (q_h,1,1)$, $(q_s,0) \rightarrow (q_h,0,1)$, $(q_s,\epsilon) \rightarrow (q_h,\epsilon,1)$

Příklad 3.2.3. *Nechť* Σ *je konečná množina a* $\epsilon \notin \Sigma$. *Potom pro* $a_1a_2...a_n \in \Sigma^*$ *existuje stroj, který na libovolném vstupu ze* Σ^* *definuje výstup* $a_1a_2...a_n$. *Tímto strojem je* $T = (A,\Sigma,Q,\delta,q_h,p_{a_1a_2...a_n},\epsilon)$, *kde* $A = \Sigma \cup \{\epsilon\}$,
$Q = (p_{a_1a_2...a_n},p_{a_2a_3...a_n},...,p_{a_n},p_\o,q_h)$ *a* δ *je určené následujícími vztahy:*
$(p_{a_ia_{i+1}...a_n},b) \rightarrow (p_{a_{i+1}...a_n},a_i,1)$, *pro všechna* $i = 1,2,3,...,n-1$ *a* $b \in A$
$(p_{a_n},b) \rightarrow (p_\o,a_n,1)$, *pro každé* $b \in A$
$(p_\o,b) \rightarrow (p_\o,\epsilon,1)$, *pro každé* $b \in \Sigma$
$(p_\o,\epsilon) \rightarrow (q_h,\epsilon,-1)$

Turing své poznatky publikoval roku 1936 v článku (Turing, 1937) (On computable Numbers and Entscheidnugsproblem). Velice brzy poté se ukázalo, že definice Turinga a Churche byly ekvivalentní. To Churche motivovalo k formulování hypotézy. Ta je dnes známá pod pojmem Church-Turingova teze. V ní stálo, že náš intuitivní pojem toho, co je algoritmicky řešitelné, splývá s tím, co je řešitelné Turingovým strojem. Poznamenejme, že dodnes nebyl vynalezen počítač, který by toho zvládnul vyřešit víc než ten jednoduchý model stroje, matematicky zkonstruovaný roku 1936 mladým nadějným studentem, Alanem Turingem.
Než se pustím do konstrukce některých Turingových strojů, tak ještě formalizujme pojem isomorfismu Turingových strojů. Účelem je sloučit dohromady stroje, jejichž výpočet probíhá stejně, až na přejmenování abecedy a vnitřních stavů.

Definice 3.2.7. *Nechť* $T = (A,\Sigma,Q,\delta,q_h,q_s,\epsilon)$ *a* $S = (B,\Pi,H,\gamma,p_h,p_s,\omega)$ *jsou Turingovy stroje. Řekneme, že* T *je isomorfní s* S *nebo že* T *je stejný jako* S *až na přejmenování prvků, pokud* $\Sigma = \Pi$ *a pokud existují bijekce* $\psi : Q \rightarrow H$, $\varphi : A \rightarrow B$ *takové, že*

$$\psi(q_s) = p_s \ , \ \psi(q_h) = p_h \ , \ \varphi(\epsilon) = \omega \ , \ \varphi(a) = a, \forall a \in \Sigma$$

a pro všechna $a^{(1)},a^{(2)} \in A$, $q^{(1)},q^{(2)} \in Q$ *a* $b^{(1)},b^{(2)} \in B$, $h^{(1)},h^{(2)} \in H$ *splňující*

$$\varphi(a^{(1)}) = b^{(1)} \ , \ \psi(q^{(1)}) = h^{(1)}$$

a vztahy

$$\delta(q^{(1)},a^{(1)}) = (q^{(2)},a^{(2)},j) \ , \ \gamma(h^{(1)},b^{(1)}) = (h^{(2)},b^{(2)},k)$$

, platí

$$\varphi(a^{(2)}) = b^{(2)} \ , \ \psi(q^{(2)}) = h^{(2)} \ , \ j = k$$

Tento fakt značíme $T \overset{(\varphi,\psi)}{\simeq} S$, *nebo jenom* $T \simeq S$. *Dvojici zobrazení* (φ,ψ) *navíc nazveme isomorfismem* T *s* S.

Lemma 3.2.1. *Relace být stejný až na přejmenování prvků pro Turingovy stroje je ekvivalencí definovanou na množině všech Turngových strojů.*

Důkaz. Pro libovolné stroje T_1,T_2,T_3 není těžké ověřit následující:

1. reflexivita: Platí, že $T_1 \overset{(id,id)}{\simeq} T_1$
2. tranzitivita: Nechť $T_1 \overset{(\varphi_1,\psi_1)}{\simeq} T_2$, $T_2 \overset{(\varphi_2,\psi_2)}{\simeq} T_3$, potom $T_1 \overset{(\varphi_1\circ\varphi_2,\psi_1\circ\psi_2)}{\simeq} T_3$
3. symetrie: Pokud $T_1 \overset{(\varphi,\psi)}{\simeq} T_2$, pak $T_2 \overset{(\varphi^{-1},\psi^{-1})}{\simeq} T_1$

□

Značení. *Třídu ekvivalence danou vztahem býti stejný až na přejmenování prvků pro stroj T označíme jako $[T]$.*

Definice 3.2.8. *Řekneme, že stroje $T = (A,\Sigma,Q,\delta,q_h,q_s,\epsilon)$ a $S = (B,\Pi,H,\gamma,p_h,p_s,\omega)$ se shodují na vstupu, pokud $\Pi = \Sigma$, $D(T) = D(S)$ a pro každé $a \in D(T)$*

$$T(a) = S(a).$$

Lemma 3.2.2. *Isomorfní Turingovy stroje se shodují na výstupu.*

Důkaz. Nechť $T = (A,\Sigma,Q,\delta,q_h,q_s,\epsilon)$ a $S = (B,\Pi,H,\gamma,p_h,p_s,\omega)$ jsou stejné až na přjmenování prvků. Potom existují (φ, ψ), že $T \overset{(\varphi,\psi)}{\simeq} S$ a platí $\Pi = \Sigma$. Chceme ukázat, že $D(T) = D(S)$ a $\forall a \in D(T), T(a) = S(a)$. Buď tedy $a \in D(T)$ a položme $K_1 = (q_s,a,1)$ a $L_1 = (p_s,a,1)$. Dále mějme $K_1 \rightarrow K_2 \rightarrow ... \rightarrow K_n$ řetězec konfigurací na stroji T a $L_1 \rightarrow L_2 \rightarrow ... \rightarrow L_n$ řetězec konfigurací na stroji S. Tvrdím, že pro každu $i \in \{1,2,...,n\}$ platí $\varphi(a_i) = b_i,\psi(q^{(i)}) = h^{(i)}$ a $j^{(i)} = k^{(i)}$, kde $K^{(i)} = (q^{(i)},a^{(i)},j^{(i)})$, $L^{(i)} = (h^{(i)},b^{(i)},k^{(i)})$. Ukažme to indukcí.

- Z definice izomorfismu platí, že $\psi(q_s) = p_s$ a jelikož $a \in \Sigma = \Pi$, tak $\varphi(a) = a$
- Z I.P. plyne, že $\varphi(a^{(i-1)}) = b^{(i-1)}$, $\psi(q^{(i-1)}) = h^{(i-1)}$ a $j^{(i-1)} = k^{(i-1)}$. Z definice isomorfismu a faktu, že $\delta(a^{(i-1)},q^{(i-1)}) = (a^{(i)},q^{(i)},l)$ pro nějaké $l \in \{1,-1\}$ a $\delta(h^{(i-1)},b^{i-1}) = (h^{(i)},b^{(i)},v)$ pro nějaké $v \in \{1,-1\}$, plyne, že $\varphi(a^{(i)}) = b^{(i)}$, $\psi(q^{(i)}) = h^{(i)}$ a $l = v$. Ovšem odsud $j^{(i)} = \max\{j^{(i-1)} + l,1\} = \max\{k^{(i-1)} + v,1\} = k^{(i)}$. A to jsme chtěli.

Jenomže z definice isomorfismu je K_n koncovou konfigurací právě tehdy, když je L_n koncovou konfigurací. Je tomu tak, neboť obraz koncového stavu T při zobrazení ψ je koncovým stavem S a naopak to platí z faktu, že ψ je prosté. Odsud tedy $D(T) = D(S)$. Protože navíc φ je bijekce fixující Σ, tak se po vynechání prvků mimo Σ řetězce $a^{(n)}$,$b^{(n)}$ rovnají. Tedy se rovnají i dané výstupy a tím jsme hotovi. □

Je dobré poznamenat, že o isomorfních strojích toho lze říct mnohem víc než jenom to, že definují stejné funkce. Pokud isomorfní stroje na daném vstupu skončí, tak takto učiní po stejném počtu kroků výpočtu, v každém kroku je jejich páska stejná až na přejmenování prvků, atp. Minulý důkaz vše dobře ilustruje.

3.2.2 Konstrukce Turingových strojů

Mnoha lidem možná na první pohled přijde, že Turingovy stroje toho moc neumí. Také se jedná o relativně jednoduchý model výpočtu. Church-Turingova teze se proto zdá být poměrně neintuitivní. Přesto se většina publikací formálním konstrukcím strojů vyhýbá. Důkaz toho, že existuje stroj, který řeší nějaký problém, redukují na jakýsi algoritmický popis práce tohoto stroje. Hlavním důvodem je bezesporu čitelnost. Kdyby autoři čtenáři předložili výčet vnitřních stavů, znaků a nějaký předpis transformační funkce, pak by se v tom u nějakých složitějších předpisů jistě ztratil. Problém ovšem je, že není úplně jasné, co do takového algoritmického popisu zahrnout, aby vše bylo stále formálně korektní. V literatuře se obvykle nějaké vymezení a odůvodnění toho, co je možné v popisu vynechat, opomíjí. O co já se v této kapitole pokusím, je tedy nastavit a odůvodnit pro zbytek této práce nějaké jasné mantinely toho, jak Turingovy stroje konstruovat a v neposlední řadě se pokusím ukázat, že Turingovy stroje jsou mnohem silnějším modelem, než se na první pohled zdá. Důsledkem tohoto všeho ovšem je, že následující podkapitola bude poměrně technická. Ty stroje, které zde zkonstruujeme, budou odkazovány zejména v části zabývající se univerzálními Turingovými stroji. Čtenář, který důvěřuje Church-Turingově tezi, ovšem může tuto podkapitolku přeskočit a vracet se k ní například pouze, když odsud budou daná lemmata odkazovaná ve zbytku této práce.

Lemma 3.2.3. *(Posuvný stroj, P) Nechť Σ je konečná množina a X,Y,$C \in \Sigma$, potom existuje stroj T se vstupní abecedou Σ, který na libovolném vstupu*

$$\alpha\ X\ \beta\ Y$$

kde znaky X,Y nejsou obsaženy v řetězcích α,$\beta \in \Sigma^*$, dává výstupní konfiguraci s řetězcem

$$\alpha\ X\ C\ \beta\ Y$$

(tzn. T posune vstupní řetězec začínající za X a na uvolněné místo vloží C)

Důkaz. Nejprve popišme, jak bude stroj T pracovat algoritmicky. T začne výpočet v počáteční konfiguraci krokem 1., se vstupem na pásce a s hlavou nastavenou na jeho prvním prvku:

1. T má nastavenou hlavu na nějakém prvku A. Máme dvě možnosti. Jestliže
   (a) $A \neq X$, pak T posune hlavu o jeden prvek doprava a pokračuje krokem 1.
   (b) $A = X$, pak si T zapamatuje X, vloží na jeho místo C, posune hlavu o jeden prvek doprava a pokračuje krokem 2.
2. Hlava stroje T je nastavena na prvku B a stroj si pamatuje nějaký znak A. T vloží A místo B.
   (a) Jestliže $A \neq Y$, pak T místo B vloží A, posune hlavu o jeden prvek doprava a pokračuje krokem 2.
   (b) Jestliže $A = Y$, pak T místo B vloží A a skončí práci.

Zbývá definovat stroj $T = (A,\Sigma,Q,\delta,q_h,q_s,\epsilon)$ formálně. Nechť Σ je jako v zadání. Potom položíme $A = \Sigma \cup \{\epsilon\}$, $Q = \{q_s,q_h\} \cup \bigcup\limits_{a\in\Sigma} p_a$. Uvědomme si, že protože A je konečná, pak i Q je konečná. Tedy vše je formálně korektní. Nakonec je tedy třeba definovat zobrazení δ. To je určeno následujícím diagramem.

1. $(q_s,a) \to (q_s,a,1), a \neq X$, $(q_s,X) \to (p_X,C,1)$
2. (a) $(p_a,b) \to (p_b,a,1), a \neq Y$
   (b) $(p_Y,b) \to (q_h,Y,1)$

☐

Příklad 3.2.4. *Ilustrujme práci stroje z lemmatu 3.2.4 na příkladu pro* $\alpha = 1$, $\beta = 45$, $X = \#$, $Y = *$, $C = 7$

$\overset{\downarrow}{1}\#45 * \epsilon\epsilon...||q_s$

$1\overset{\downarrow}{\#}45 * \epsilon\epsilon...||q_s$

$17\overset{\downarrow}{4}5 * \epsilon\epsilon...||p_\#$

$17\#\overset{\downarrow}{5} * \epsilon\epsilon...||p_4$

$17\#4 \overset{\downarrow}{*} \epsilon\epsilon...||p_5$

$17\#45\overset{\downarrow}{\epsilon}\epsilon...||p_*$

$17\#45 * \overset{\downarrow}{\epsilon}...||q_h$

V této konstrukci jsme udělali pár triků, které jsou obecně použitelné (a používané). Vstup se často prokládá nějakými speciálními znaky, v našem případě X,Y. Tyto znaky obvykle plní funkci jakéhosi oddělovače sektorů pracovní pásky a umožňují stroji se na pásce orientovat. V předchozím lemmatu například X odděluje sektor, který má stroj posunout od sektoru, který má zůstat nezměněn. Dalším poměrně očividným důsledkem je, že se není třeba při algoritmickém popisu práce stroje zabývat tím, na jakém políčku je ve kterém kroku nastavena hlava. Pokud je pozice totiž jednoznačně popsatelná pomocí nějakého prvku (nebo sekvence nalezení prvků), pak je poměrně triviální se na toto políčko dostat. Stroj se prostě požadovaným směrem posnuje dokud na daný znak nenarazí. V našem případě tomu například odpovídá vnitřní stav q_s, ve kterém stroj hledá znak Y. Potom by na to mohl navázat dalším stavem $q_s^{(2)}$ ve kterém by zase našel jiný prvek a takto by pokračoval. Jediné, co je podstatné, je, aby daných hledání bylo konečně mnoho. Posledním důsledkem je, že stroj je schopen načíst a zapamatovat si nějaký prvek. V předchozí konstrukci tuto roli plnil stav p_a, kde a v dolním indexu je daný znak abecedy. Tím stroj může při rozhodování o dalším postupu pracovat s více než jen jednou proměnnou v podobě znaku, nad kterým se v daném momentu nachází jeho hlava. Není už velkým myšlenkovým krokem, že stroj takto umí načíst řetězec z nějaké konečné množiny řetězců.
Poznamenejme ještě, že stavy q_s,p_a poměrně dobře ilustrují intuici z úvodu minulé podkapitoly, tedy to, že jednotlivé stavy odpovídají jakémusi „stavu mysli“ tohoto stroje. Když je stroj ve stavu q_s, tak je ve stavu kdy, prohlíží jednotlivé znaky a hledá Y. Ve stavu p_a si stroj pamatuje znak a, který se chystá vložit na následující políčko.

Lemma 3.2.4. *(Posuvný stroj, L) Nechť Σ je konečná množina a mějme X,Y,$Z \in \Sigma$, potom existuje stroj T, který na libovolném vstupu*

$$X \ \alpha \ Y \ \beta \ Z \in \Sigma^+$$

kde znaky X,Y,Z nejsou obsaženy v řetězcích α,$\beta \in \Sigma^+$, dává výstupní konfiguraci s řetězcem

$$X \ Y \ \beta \ Z \ \bar{\epsilon},$$

pro nějaké $\bar{\epsilon} \in \{\epsilon\}^*$.
(tzn. T posune celý řetězec ohraničený Y a Z doleva vedle prvku X)

Důkaz.

1. T projde vstup zleva doprava, najde Y a všechny prvky (kromě X,Y) po cestě nahradí ϵ. Pokračuje krokem 2.
2. V aktuální konfiguraci je nějaký řetězec $a_1 a_2 ... a_n \epsilon\epsilon ... \epsilon b_1 b_2 ... b_m$, kde a_i,$b_j \in \Sigma$. T prohodí b_1 s ϵ z levého konce řetězce ϵ. Tedy transformuje pásku do tvaru $a_1 a_2 ... a_n b_1 \epsilon\epsilon ... \epsilon b_2 ... b_m$. Pokračuje krokem 3.
3. V aktuální konfiguraci je nějaký řetězec $a_1 a_2 ... a_n \epsilon\epsilon ... \epsilon b_1 b_2 ... b_m$, kde a_i,$b_j \in \Sigma$.
   (a) Pokud $b_1 = Z$, pak T prohodí Z s ϵ z levého konce řetězce ϵ a skončí práci.
   (b) Pokud $b_1 \neq Z$, pak pokračuje krokem 2.

Opět uvedeme diagram definující funkci δ. Pokud není uvedeno jinak, pak a,$b \in \Sigma$ jsou libovolné

1. $(q_s,a) \to (s_\epsilon,a,1)$, $(s_\epsilon,a) \to (s_\epsilon,\epsilon,1)$, $a \neq Y$, $(s_\epsilon,Y) \to (n_Y,\epsilon,-1)$
2. $(n_b,\epsilon) \to (n_b,\epsilon,-1)$, $(n_b,a) \to (v_b,a,1)$,$a \neq \epsilon$, $(v_b,a) \to (l,b,1)$
3. $(l,\epsilon) \to (l,\epsilon,1)$,
   (a) $(l,Z) \to (k,\epsilon,-1)$, $(k,\epsilon) \to (k,\epsilon,-1)$, $(k,a) \to (v,a,1)$, $a \neq \epsilon$, $(v,a) \to (q_h,Z,1)$
   (b) $(l,b) \to (n_b,\epsilon,-1)$, $b \neq Z,\epsilon$

□

V některých řešeních je nutné, aby si stroj pamatoval, která políčka už navštívil, zkopíroval, zkontroloval atp. Protože jich ovšem pro některé problémy může být libovolně mnoho, tak si stroj nějak dané pozice na pásce musí označit. Pro další postup je ovšem často potřeba, aby si nesmazal informaci o tom, jaké prvky se tam před zmiňovaným označením nacházely. Reálně se to dá řešit tak, že se pro každý znak a vybere nějaký symbol, který slouží jako jeho označené dvojče. Jak jej označíme je v podstatě jedno. Jediné co je potřeba, je, aby dané označení bylo jednoznačné a mělo prázdný průnik se zbytkem pracovní abecedy. V příkladu za následujícím lemmatem jsme pro ilustraci vybrali „podtržení“ a „otečkování“. V tom případě by označené dvojče a bylo $\dot{a}$ resp. $\underline{a}$

Lemma 3.2.5. *(Stroj shody) Nechť Σ je konečná množina a $X,Y,Z \in \Sigma$, potom existuje stroj $T = (A,\Sigma,Q,\delta,q_h,q_s,\epsilon)$, který na libovolném vstupu:*

$$\omega = X \ \alpha \ Y \ \beta \ Z \in \Sigma^+,$$

kde znaky X,Y,Z nejsou obsaženy v řetězcích $\alpha,\beta \in \Sigma^+$, dává výstupní konfiguraci (ω,q_h,j), kde

$$j := \begin{cases} \textit{souřadnice začátku nejlevějšího podřetězce } \alpha\textit{, který je roven } \beta \\ \textit{souřadnici } Y\textit{, jestliže takový podřetězec neexistuje} \end{cases}$$

Důkaz. Uvažujme nějaké dvě libovolné množiny B, C, kde $|B| = |A| = |C|$ jsou po dvou disjunktní a zvolme nějaká pevná prostá zobrazení $\varphi : A \to C$, $\psi : A \to B$. O obrazech φ, ψ budeme mluvit jako o označených prvcích. T pracuje následovně:

1. Nechť a je první znak na pásce. T místo něj vloží $\psi(a)$ a pokračuje krokem 2.
2. Na pásce je právě jeden znak c označený ψ. Nechť a je první znak neoznačený φ za c (doprva) a b je první neoznačený prvek φ za Y. T vloží místo a znak $\varphi(a)$. Pokud navíc
   (a) $a = b$, pak T označí pomocí φ i b a pokračuje krokem 2.
   (b) $a \neq b$, pak jestliže
      i. $b = Z$, pak T pokračuje krokem 4.
      ii. $b \neq Z$, pak T pokračuje krokem 3.
3. T všechny prvky, které jsou na pásce označeny φ odznačí. Nechť a je pravý soused prvku označeného ψ. Pokud
   (a) $a = Y$, pak T odznačí znak označený ψ a skončí s hlavou nastavenou na Y.
   (b) $a \neq Y$, pak T označení ψ posune o jeden prvek doprava. Pokračuje krokem 2.
4. T vše označené φ odznačí, prvek označený ψ také odznačí a skončí s hlavou na jeho pravém sousedovi.

Formálně stroj $T = (A,\Sigma,Q,\delta,q_h,q_s,\epsilon)$ definujeme následovně. Nechť $\varphi(\Sigma) := \{\varphi(a) : a \in \Sigma\},\psi(\Sigma) := \{\psi(a) : a \in \Sigma\}$ a položme $A = \Sigma \cup \varphi(\Sigma) \cup \psi(\Sigma)$, $Q = \{q_s,q_h,z^{(2)},z^{(3)},z^{(4)},o,v\} \cup \bigcup\limits_{a\in\Sigma} n_a^{noz} \cup \bigcup\limits_{a\in\Sigma} n_a^{(Y)}$. Funkce δ je daná následujícím diagramem.

1. $(q_s,a) \to (z^{(2)},\psi(a),1)$
2. $(z^{(2)},a) \to (n_a^{(Y)},\varphi(a),1)$, $a \notin \varphi(\Sigma)$, $(z^{(2)},\varphi(a)) \to (z^{(2)},\varphi(a),1)$, $a \in \Sigma$
   $(n_a^{(Y)},b) \to (n_a^{(Y)},b,1)$, $b \neq Y$, $(n_a^{(Y)},Y) \to (n_a^{noz},Y,1)$,
   $(n_a^{noz},\varphi(b)) \to (n_a^{noz},\varphi(b),1)$
   (a) $(n_a^{noz},a) \to (n^{(3)},\varphi(a),-1)$, $(n^{(3)},a) \to (n^{(3)},a,-1), a \neq \psi(\Sigma)$
   $(n^{(3)},\psi(a)) \to (z^{(2)},\underline{a},1), a \in \Sigma$

    (b) i. $(n_a^{noz},Z) \to (z^{(4)},Z,-1)$
        ii. $(n_a^{noz},b) \to (z^{(3)},a,-1), b \notin \varphi(\Sigma), b \neq Z,a$

3. $(z^{(3)},\varphi(a)) \to (z^{(3)},a,-1), a \in \Sigma$, $(z^{(3)},a) \to (z^{(3)},a,-1), a \in \Sigma$
   $(z^{(3)},\psi(a)) \to (o,a,1), a \in \Sigma$
    (a) $(o,Y) \to (v,Y,1)$, $(v,a) \to (q_h,a,-1)$
    (b) $(o,a) \to (z^{(2)},\psi(a),1), a \neq Y$
4. $(z^{(4)},\varphi(a)) \to (z^{(4)},a,-1)$, $(z^{(4)},\psi(a)) \to (q_h,a,1)$

□

Příklad 3.2.5. *Následující tabulka ilustruje práci stroje z lemmatu 3.2.5* $\alpha = 10110$ *a* $\beta = 11$. *Jako označení* $\varphi(a)$ *bereme* $\dot{a}$ *a jako* $\psi(a)$ *bereme* $\underline{a}$. *Nejedná se o znázornění řetězce konfigurací (krok za krokem). Některé konfigurace jsou vynechány, souřadnice hlavy je až na jednu výjimku také opomenuta. Celkový výpis by zabral zbytečně moc místa.*
$X10110Y11Z$
$\underline{X}\dot{1}0110Y\dot{1}1Z$
$\underline{X}\dot{1}\dot{0}110Y\dot{1}1Z$
$X\underline{1}0110Y11Z$
$X\underline{1}\dot{0}110Y11Z$
$X1\underline{0}110Y11Z$
$X1\underline{0}\dot{1}10Y\dot{1}1Z$
$X1\underline{0}\dot{1}\dot{1}0Y\dot{1}\dot{1}Z$
$X1\underline{0}\dot{1}\dot{1}\dot{0}Y\dot{1}\dot{1}Z$
$X10\overset{\downarrow}{1}10Y11Z$

Vidíme, že diagram určující funkci δ už začíná být poměrně složitý. Představme si, jak dlouhý by asi byl výpis konstrukce stroje, který by realizoval nějaký netriviální algoritmus. Pro čtenáře by taková konstrukce byla jistě nečitelná. Čeho je ovšem možné si všimnout, je, že většina položek nevyhnutelně popisuje více méně stále to samé dokola. Přesouvá hlavu z místa na místo, podle aktuálního prvku rozděluje další postup na případy, označuje prvky atp. V diskuzi za lemmatem 3.2.4 jsme naznačili, že není v algoritmickém popisu třeba se starat o to, kde se nachází hlava konstruovaného stroje. Argumentovali jsme nějakým obecným postupem, který nejprve našel jeden prvek, potom druhý, a tak pokračoval, dokud se nedostal na požadované místo. Jeden problém jsme tedy rozdělili do více menších podproblémů. Není těžké si rozmyslet, že každému z těchto dílčích hledání odpovídá nějaký stroj. Otázka zní jak obecně lze tento trik používat. Neboli jak a za jakých podmínek lze na vstup postupně pouštět stroje z nějaké konečné množiny, které nám budou upravovat vnitřní konfigurace do konečné podoby. V takových případech by šel libovolný stroj popsat pomocí již zkonstruovaných a triviálně zkonstruovatelných strojů. Nadále bychom se tedy mohli vyvarovat formální konstrukci a práci stroje popsat čistě algoritmicky pomocí toho, co je triviální a toho, co je již zkonstruované. Použitelné mantinely toho, jak je možné tento postup požívat nastavují následující dvě lemmata.

Lemma 3.2.6. *(Existence sekvenčního stroje) Ať $T = (A,\Sigma,Q,\delta,q_h,q_s,\epsilon)$, $S = (B,\Pi,H,\gamma,p_h,p_s,\epsilon)$, kde $A \supseteq B$ a $Q \cap H = \emptyset$. Nechť navíc (b_a,q_h,i_a), je koncová konfigurace stroje T při vstupu $a \in D(T)$. Pro ta $a \in D(T)$, pro která existuje koncová konfigurace stroje S při vstupní konfiguraci (b_a,p_s,j_a), ji označme (c_a,p_h,i_a). Potom existuje stroj U, který na libovolném vstupu $a \in D(T)$, na kterém je konfigurace (c_a,p_h,j_a) definovaná, dáva výstupní konfiguraci (c_a,p_h,j_a) a na zbylých vstupech stroje T nedefinuje výstup.*
(tzn. U na libovolný vstup nejprve pustí stroj T a potom stroj S)

Důkaz. Stroj U se poměrně snadno zkonstruuje tak, že položíme
$U := (\Sigma,A_S,Q_1 \cup Q_2 \cup v,\xi,q_s,p_h,\epsilon)$, kde $v \notin Q_1 \cup Q_2$ je libovolné a ξ splňuje:

$$\xi := \delta \cup \gamma \cup \bigcup_{a\in A_S} ((q_h,a),(v,a,1)) \cup \bigcup_{a\in A_S} ((v,a),(p_s,a,-1)),$$

tzn.ξ je rozšířené o vztahy $(q_h,a) \to (v,a,1)$, $(v,a) \to (p_s,a,-1)$, které víceméně jen dostanou hlavu na správné místo. Nakonec si uvědomme, že protože Q_1 a Q_2 jsou konečné, pak také $Q_1 \cup Q_2 \cup \{v\}$ je konečná. Tím je konstrukce hotová □

Poznamenejme ještě, že disjunktnost množin stavů ve znění předchozího lemmatu není problém. Při nějakých aplikacích, lze bez újmy na obecnosti předpokládat, neboť při nejhorším můžeme jednotlivé prvky prostě přejmenovat.

Lemma 3.2.7. *(Spouštěcí stroj)Nechť A je konečná množina, $X,Y,Z \in A$, $A_T \subseteq A \smallsetminus \{X,Y,Z\}$ a T je stroj s pracovní abecedou A_T a prázdným prvkem $\epsilon \in A_T$. Potom existuje stroj U, který na libovolném vstupu*

$$\gamma\ X\ \alpha\ Y\ \beta\ Z$$

kde znaky X,Y,Z nejsou obsaženy v řetězcích $\gamma,\alpha,\beta \in A_T^*$ dá výstupní konfiguraci s řetězcem

$$\gamma\ X\ \alpha_T\ Y\ \beta\ Z$$

kde α_T je řetězec koncové konfigurace, kterou implikuje stroj T na vstupu $\alpha \in D(T)$ a nedefinuje žádný výstup, pro $\alpha \notin D(T)$.
(tzn. U pracuje jako T na úseku ohraničeném X a Y)

Důkaz. Stroj U bude pracovat s označením φ, pomocí kterého si bude pamatovat, na jakém prvku pásky má spouštěný stroj v dané chvíli nastavenou hlavu.

1. U označí první prvek za X pomocí φ a zapamatuje si q_s.
2. Nechť $\varphi(a)$ je jediný označený prvek na pásce a U si pamatuje stav q. Jestliže
   (a) $a = Y$, potom U posune celý vstup za Y doprava a na uvolněné místo vloží $\varphi(\epsilon)$ (viz stroj. 3.2.3), odznačí Y. Pokračuje krokem 2.
   (b) $a = X$, potom U posune označení o jeden prvek doprava a pokračuje krokem 2.
   (c) $a \neq X,Y$ a $\delta(q,a) = (g,b,i)$, kde
      i. $g = q_h$, pak U místo $\varphi(a)$ vloží b a skončí práci
      ii. $g \neq q_h$, pak U místo $\varphi(a)$ vloží b, označení vloží o $i \in \{1,-1\}$ prvků doprava a pokračuje krokem 2.

□

Uvědomme si, jak silný nástroj nám dvě předchozí lemmata dávají. Sekvenční stroj nám umožňuje postupně upravovat pásku způsobem, který připomíná volání knihovních funkcí a spouštěcí stroj nám umožňuje, aby dané stroje pracovaly na nějakém izolovaném úseku pásky a neměnily zbytek.
Ještě poznamenejme, že jsme zatím v každém lemmatu pracovali s nějakými specifickými znaky neobsaženými na zbytku pásky. To se zdá být pro sekvenční stroj poměrně svazující. Stroj U, který bude dané stroje volat, si ovšem může tyto oddělovací znaky sám vytvořit. Udělá to tak, že tam, kde si to algoritmus žádá, dané prvky prostě nějak specificky označí. A ve chvíli, kdy volaný stroj skončí práci, si je pak zase odznačí.
Co tedy už všechno umíme?

- Rozdělit úlohu na menší podúlohy. Ty vyřešit nějakými stroji a práci těchto strojů zřetězit.
- Spustit stroj na nějakém izolovaném segmentu pásky
- Dostat se na libovolné políčko, které je popsatelné pomocí sekvence hledání prvků
- Posunout nějaký podřetězec doprava a na uvolněné místo vložit nějaký znak
- Posunout nějaký podřetězec doleva
- Načíst nějaký prvek nebo řetězec z konečné množiny řetězců
- Specificky označovat prvky.

Všechny tyto položky použijeme v následujících dvou důležitých lemmatech.

Lemma 3.2.8. *(Substituční stroj 1) Nechť Σ je konečná množina, X,Y,$Z \in \Sigma$, $B \subseteq \Sigma^*$ je konečná množina a $\mathscr{C} : B \to \Sigma^*$ je nějaké konečné zobrazení. Potom existuje stroj T s prázdným znakem ϵ, vstupní abecedou Σ a pracovní abecedou obsahující $\Sigma \cup B$, který na libovolném vstupu*

$$X \ \alpha \ Y \ \beta \ Z,$$

kde X,Y,Z jsou znaky, který nejsou obsaženy v řetězcích α,$\beta \in \Sigma^*$, dává výstupní konfiguraci s řetězcem

$$X \ \mathscr{C}(\alpha) \ Y \ \beta \ Z \ \bar{\epsilon}$$

pro nějaké $\bar{\epsilon} \in \{\epsilon\}^*$.
(tzn. T substituuje řetězec α řetězcem $\mathscr{C}(a)$).

Důkaz.

1. U načte řetězec α mezi X a Y. Všechny jeho prvky přitom nahradí ϵ. Zapamatuje si $\mathscr{C}(\alpha)$.
2. U vypíše mezi znaky X a Y řetězec, který si pamatuje (zavolá vypisovací stroj z příkladu 3.2.3 na části izolované X a Y).

3. Nyní je na pásce řetězec $X\ \mathscr{C}(\alpha)\ \bar{\epsilon}\ Y\ \beta\ Z$, pro nějaké $\bar{\epsilon} \in \{\epsilon\}^*$
   U posune řetězec $Y\ \beta\ Z$ vedle $\mathscr{C}(\alpha)$ (viz lemma 3.2.4) a skončí práci.

□

Lemma 3.2.9. *(Substituční stroj 2) Nechť Σ je konečná množina a $W,X,Y,Z \in \Sigma$, potom existuje stroje T, který na libovolném vstupu*

$$W\ \alpha\ X\ \beta\ Y\ \gamma\ Z\ \omega\ V$$

kde znaky W,X,Y,Z,V nejsou obsaženy v řetězcích $\alpha,\beta,\gamma,\omega \in \Sigma$, dává výstup

$$W\ \alpha\ X\ \beta\ Y\ \alpha\ Z\ \omega\ V\ \bar{\epsilon}$$

pro nějaké $\epsilon \in \{\epsilon\}^*$
(tzn. T substituuje řetězec ohraničný Y,Z za ten ohraničený W,X)

Důkaz.

1. Nechť a je první neoznačený prvek za W a b je první neoznačený prvek za Y
   (a) Pokud $a = X$, potom jestliže
      i. $b = Z$, pak skončí práci
      ii. $b \neq Z$, pak T posune řetězec $Z\ \omega\ V$ doleva vedle b (viz lemma 3.2.4).
   (b) Pokud $a \neq X$, pak jestliže
      i. $b = Z$, pak T posune celý řetězec za Z o jedno políčko doprava a na uvolněné místo vloží a (viz lemma 3.2.3). Potom oba prvky označí a pokračuje krokem 1.
      ii. $b \neq Z$, pak T vloží a místo b, oba prvky označí a pokračuje krokem 1.

□

3.2.3 Univerzální Turingův stroj

V podkapitole 3.2.1 jsme mluvili o isomorfismu Turingových strojů. Zadefinovali jsme tak třídu ekvivalence [ ] „býti stejný až na přejmenování prvků“. Ukázali jsme navíc, že stroje, které jsou ekvivalentní, v podstatě pracují stejně. Každá třída ekvivalence, která je definovaná tímto isomorfismem, nám tedy dává nějaký unikátní postup. Pokud naopak nějaké dva stroje přísluší různým třídám, tak fromalizují odlišné algoritmy. Ukazuje se, že každému algoritmu lze jednoznačně přiřadit nějaký binární řetězec. To jistě v dnešní době plné počítačových programů není žádným překvapením. Pokusme se to však ukázat formálně.

Definice 3.2.9. *Nechť $T = (A,\Sigma,Q,\delta,q_h,q_s,\epsilon)$ je Turingův stroj. Potom pětici $(q,a,\tilde{q},\tilde{a},\tilde{i}) \in Q \times A \times Q \times A \times \{1,-1\}$ takovou, že $\delta(q,a) = (\tilde{q},\tilde{a},\tilde{i})$ nazveme příkazem stroje T. Množinu všech příkazů stroje T budeme značit Com_T*

Značení. *Pro každý Turingův stroj $T = (A,\Sigma,Q,\delta,q_h,q_s,\epsilon)$, kde $A = \{\epsilon,0,1,a_1,\ldots,a_n\}$ a $Q = \{q_s,q_h,q_1,\ldots,q_r\}$ uvažujme trojici kódování:*

$$\begin{array}{llllll} C_A: & 1 \to 1 & C_Q: & q_s \to 1 & C_P: & 1 \to 1 \\ & 0 \to 11 & & q_h \to 11 & & -1 \to 11 \\ & \epsilon \to 111 & & q_j \to 1^{(j+2)} & & \\ & a_i \to 1^{(i+3)} & & & & \end{array}$$

Potom pro $v = (q,a,\tilde{q},\tilde{a},\tilde{i}) \in Com_T$ definuji řetězec:

$$C^T(v) = C_Q(q)\ 0\ C_A(a)\ 0\ C_Q(\tilde{q})\ 0\ C_A(\tilde{a})\ 0\ C_P(\tilde{i})\ 0$$

Seřaďme nyní prvky $\{C^T(v)\colon v = (q,a,\tilde{q},\tilde{a},\tilde{i}) \in Com_T\}$ primárně podle délky $C_Q(q)$ a sekundárně podle délky $C_A(a)$. Takto seřazené prvky napišme za sebe a výsledný řetězec označme $\alpha(T)$.

Tvrzení 3.2.10. *Existuje jednoznačná korespondence mezi množinou*

$$\{[T] : T \text{ je binární Turingův stroj}\}$$

a nějakou podmnožinou $\{0,1\}^*$

Důkaz. Chceme ukázat, že $\alpha(T)$ jednoznačně určuje binární Turingův stroj (až na přejmenování prvků). Pro tento účel si uvědomme, že pokud dva binární Turingovy stroje $T = (A,\Sigma,Q,\delta,q_h,q_s,\epsilon)$ a $S = (B,\Pi,H,\gamma,p_h,p_s,\omega)$ dávají stejné sekvence $\alpha(T) = \alpha(S)$, pak určují také stejné množiny:

$$\{C^T(v) : v \in \mathrm{Com}_T\} = \{C^S(v) : v \in \mathrm{Com}_S\}$$

Je tomu tak neboť C^T dostanu z $\alpha(T)$ prostě tak, že tento řetězec „rozstříhnu“ za každou pátou nulou. Odsud ihned plyne, že Q s H a A s B jsou stejně početné. Definujeme-li tedy bijekce $\varphi : A \to B$ a $\psi : Q \to H$ vztahem

- $\varphi(a) := b$ pro $a \in A, b \in B$ splňují $C_A(a) = C_B(b)$
- $\psi(q) := h$ pro $q \in Q, h \in H$ splňují $C_Q(q) = C_H(h)$

potom, protože $C_A(\epsilon) = C_B(\omega) = 111$, pak $\varphi(\epsilon) = \omega$: Podobně pak $\varphi(1) = 1$, $\varphi(0) = 0$, $\psi(q_s) = p_s$, $\psi(q_h) = p_h$. Zbývá ukázat, že se φ, ψ chovají hezky vůči transformační funkci. Neboli chceme ukázat, že pokud $(q,a,\tilde{q},\tilde{a},\tilde{i}) \in Com_T$, pak $(\psi(q),\varphi(a),\psi(\tilde{q}),\varphi(\tilde{a}),\tilde{i}) \in Com_S$. To je ovšem poměrně očividné, neboť z definice φ,ψ:

$$\begin{aligned} C^T(q,a,\tilde{q},\tilde{a},\tilde{i}) &= C_Q(q)\ 0\ C_A(a)\ 0\ C_Q(\tilde{q})\ 0\ C_A(\tilde{a})\ 0\ C_P(\tilde{i})\ 0 \\ &= C_H(\psi(q))\ 0\ C_B(\varphi(a))\ 0\ C_H(\psi(\tilde{q}))\ 0\ C_B(\varphi(\tilde{a}))\ 0\ C_P(\tilde{i})\ 0 \\ &= C^S(\psi(q),\varphi(a),\psi(\tilde{q}),\varphi(\tilde{a}),\tilde{i}) \end{aligned}$$

a protože $\{C^T(v){:}v \in \mathrm{Com}_T\} = \{C^S(v){:}v \in \mathrm{Com}_S\}$, tak ihned

$$C^S(\varphi(a),\psi(q),\varphi(\tilde{a}),\psi(\tilde{q}),\tilde{i}) \in \{C^S(v) : v \in Com_S\}$$

Ale C^T, C^S jsou prostá zobrazení, proto jednoznačně $\gamma(\psi(q),\varphi(a)) = (\psi(\tilde{q}),\varphi(\tilde{a}),\tilde{i})$. Tedy ukázali jsme, že pokud $\alpha(T) = \alpha(S)$, potom

$$T \overset{(\varphi,\psi)}{\simeq} S.$$

Vyberme konečně pro každou třídu ekvivalence $[T]$, kde T je binární Turingův stroj, nějaké pevné číslo $\alpha(T)$. Takto dostaneme jednoznačnou korespondenci mezi $\{[T]\ :\ T \text{ binární Turingův stroj}\}$ a nějakou podmnožinou množiny všech konečných posloupností 0 a 1. $\square$

Definice 3.2.10.

- *Zobrazení $\psi : \{T :\ \text{binární Turingův stroj}\} \to \{0,1\}^*$ splňující, že $\psi(T_1) = \psi(T_2)$ implikuje $[T_1] = [T_2]$, pro všechny binární Turingovy stroje T_1,T_2, se nazývá deskripčním zobrazením. Prvky jeho obrazu jsou deskripční čísla.*
- *Binární řetězec $\alpha(T)$ z předchozího důkazu nazveme Turingovým číslem stroje T.*

Příklad 3.2.6. *V příkladu 3.2.2 jsme definivali stroj T, který pásku ponechal ve stejném stavu a skončil práci. Byl definován vztahy*

$$(q_s,1) \to (q_h,1,1), (q_s,0) \to (q_h,0,1), (q_s,\epsilon) \to (q_h,\epsilon,1)$$

Jeho Turingovo číslo je řetězec:

$$\alpha(T) = (101011010101011011011010101110110111010)$$

Není těžké si uvědomit, že $\alpha(T)$ je z definice deskripčním číslem T. Poměrně jednoduchým důsledkem jeho existence je to, že existuje pouze spočetně mnoho „různých“ binárních algoritmů.

Důsledek 3.2.11. *Množina $M = \{[T] :\ \text{binární Turingův stroj}\}$ je spočetná.*

Důkaz. Vezmeme-li Turingovo číslo z tvrzení 3.2.10 a pro každou třídu ekvivalence $[T]$ vybere pouze jedno $\alpha(T)$, pak dostaneme jednoznačnou korespondenci mezi M a nějakou podmnožinou $\{0,1\}^*$. Ta je ovšem nutně spočetná. $\square$

Poznámka 3.2.12. *Když Turingovy stroje seřadíme lexikograficky podle jejich Turingova čísla, potom nám vznikne posloupnost* T_1,T_2,.... *Index i Turingova stroje T, že* $T_i = T$ *se občas nazývá Gödelovým indexem T a posloupnost* T_1,T_2,... *standartní enumerací Turingových strojů. Ukazuje se dokonce, že zobrazení* $\alpha(T_i) \to (i)_2$ *je řešitelné (tzn. existuje Turingův stroj S, že* $S(\alpha(T_i)) = (i)_2$*) .*

Deskripční čísla mají jednu nesmírně důležitou aplikaci. Jak za chvilku ukážeme, tak mezi všemi binárními Turingovými stroji existuje jistá podskupina strojů dostatečné složitosti, které jsou schopny napodobovat chod libovolného jiného stroje jenom za poskytnutí jeho deskripčního čísla a vstupu. Tyto stroje se v dnešní literatuře nazývají univerzálními Turingovými stroji. Paralela univerzálních strojů s dnešními programovatelnými počítači je poměrně očividná. Tento fakt je o to překvapivější, když si uvědomíme, v jaké době Turing tuto teorii vymýšlel. Ve třicátých letech již nějaké mechanické počítače sice existovaly, známým příkladem je superpočítač IBM, ale ve chvíli, kdy chtěl člověk změnit vykonávanou funkci takového přístroje, tak musel ručně zasáhnout do jeho fyzické konfigurace. Alan Turing tedy v jistém smyslu předpověděl programovatelné počítače. To jenom ukazuje, jak napřed oproti své době ve skutečnosti byl.
Jelikož univerzální stroje přijímají informaci skrz dva binární řetězce a Turingovy stroje mají jenom jeden vstup, tak je třeba abychom mu je předávali skrz nějakou párovací funkci. Na tuto funkci klademe dva požadavky. V podstatě libovolné zobrazení, které je splňuje pro náš účel postačí. V první řadě z určitých důvodů chceme, aby délka této párovací funkce nezávisela na vstupu. Dále je třeba, aby existoval nějaký stroj, který je schopen z tohoto zápisu opět oba dílčí řetězce oddělit. Jinými slovy chceme, aby existoval stroj, který párovací funkci bude schopen interpretovat. Z těchto důvodů jsem vybral následující párovací funkci:

Definice 3.2.11. *Definujeme zobrazení* $(\ ,\ ) : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*$ *následovně,* $\forall \alpha,\beta \in \{0,1\}^*$

$$(\alpha,\beta) := 1^{(\ell(\alpha))}\ 0\ \alpha\ \beta,$$

Ukažme, že tato funkce jde opravdu interpretovat nějakým Turingovým strojem.

Tvrzení 3.2.13. *Existuje stroj T, který na vstupu* $(\alpha,\beta), \alpha, \beta \in \{0,1\}^*$ *dá výstupní konfiguraci s řetězcem*

$$\#\ \alpha\ *\ \beta\ \bar{\epsilon}$$

pro nějaké $\bar{\epsilon} \in \{\epsilon\}^*$.

Důkaz.

1. T posune vstup a na uvolněné místo vloží #. Pokračuje krokem 2.
2. Nechť a je první neoznačený prvek za #. Jestliže
    (a) $a = 1$, tak T označí a a první neoznačený prvek za nejbližší 0 od a doprava. Pokračuje krokem 2.
    (b) $a = 0$, tak T posune řetězec začínající na pravém sousedovi a doleva vedle #(viz lemma 3.2.4). Pokračuje krokem 3.

3. Na pásce je nyní řetězec '$\#\ \alpha\ \beta$', kde $\beta \in \{0,1\}^*$, $\alpha \in \{\dot{0},\dot{1}\}^*$. T posune β o jeden prvek doprava a na uvolněné místo vloží $*$. Potom odznačí všechny prvky α a skončí práci.

$\square$

Úmluva. *Abychom se neuzávorkovali, tak pro libovolný binární Turingův stroj* T *a každé* $x,y \in \{0,1\}^*, (x,y) \in D(T)$ *označíme* $T(x,y) := T((x,y))$

Definice 3.2.12. *(Univerzální Turingův stroj) Binární Turingův stroj* U *nazveme univerzální, pokud existuje deskripční zobrazení* P *takové, že pro každý binární Turingův stroj* T *a* $x \in D(T)$ *platí*

$$U(P(T),x) = T(x)$$

a pro každé $x \in \{0,1\}^*$*, že* $x \notin D(T)$ *platí* $(P(T),x) \notin D(U)$.

Poznámka 3.2.14. *Jestliže* U *je univerzální stroj, pak pro každý binární Turingův stroj* T *existuje* $t \in \{0,1\}^*$*, že pro každé* $x \in D(T)$

$$U(t \circ x) = T(x)$$

Důkaz. Hledané t je určené jako $t := 1^{\ell(P(T))}\ 0\ P(T)$. $\square$

Věta 3.2.15. *Existuje univerzální Turingův stroj.*

Důkaz. Nyní se pokusíme zkonstruovat univerzální Turingův stroj U. Prvně si uvědomme, že univerzální stroje umí simulovat všechny binární Turingovy stroje na nějakém fixním prefixu (nezávisejícím na vstupu - viz lemma 3.2.14). U tedy musí mít nějaký obecný mechanismus, jak řetězit jednotlivé konfigurace libovolného stroje jehož práci bude napodobovat. Problém je, že ať zvolíme pracovní abecedu a množinu vnitřních stavů libovolně velké, vždy bude existovat nějaký stroj, který je bude mít větší. Aby si tedy U mohl pamatovat aktuální konfiguraci takového stroje, tak si ji bude muset nějak zakódovat na pásku. V našem případě bude U pracovat s příslušnými C_A a C_Q kódy. Dalším problémem je, jak má stroj U vědět, které konfigurace na sebe u daného stroje bezprostředně navazují. Tuto informaci očividně musí stroji U nějak předat prefix příslušný simulovanému stroji. V našem případě pro stroj T zvolíme program $P(T) := \alpha(T)$ a tedy prefix $1^{\ell(\alpha(T))}\ 0\ \alpha(T)$. Jelikož $\alpha(T)$ kóduje transformační funkci příslušného stroje T, tak obsahuje všechnu informaci o tom, jak na sebe všechny jeho konfigurace navazují.

Páska stroje U bude tedy rozdělena do tří sektorů:

Kód stroje || Buffer || Simulační páska

Na začátku každého kroku simulačního cyklu stroje $T = (A,\Sigma,Q,\delta,q_h,q_s,\epsilon)$ budou jednotlivé sektory obsahovat následující:

- V kódu stroje bude uloženo číslo $\alpha(T)$ Pro usnadnění orientace na pásce budou jednotlivé 0 nahrazeny speciálními znaky. Místo toho aby tam byl každý příkaz uložen v původním tvaru, tzn. jako
  '$C_Q(q)\ 0\ C_A(a)\ 0\ C_Q(\tilde{q})\ 0\ C_A(\tilde{a})\ 0\ C_P(\tilde{i})\ 0$'
  tak si jej U přepíše na
  '$C_Q(q)\ ,\ C_A(a) \rightarrow C_Q(\tilde{q})\ ,\ C_A(\tilde{a})\ ;\ C_P(\tilde{i})\ |$'

- V Bufferu bude mít U uložený pár $(q,a) \in Q \times A$ aktuální konfigurace stroje T. Protože U bude hledat shodu tohoto vstupu s nějakým vstupem δ v kódu stroje, tak si jej U bude udržovat zakódovaný ve tvaru:
  '$C_Q(q)$, $C_A(a) \rightarrow$'.
- Simulační páska bude obsahovat řetězec $\alpha \in A^*$ aktuální konfigurace stroje T zakódovaný tak, že jestliže $\alpha = a_1 a_2 ... a_n$, kde $a_i \in A^*, i = 1,2,...,n$, pak na simulační pásce bude obsaženo: $C_A(a_1)|C_A(a_2)|...|C_A(a_n)|$. Ještě je potřeba, aby měl U na pásce nějak zakódovanou souřadnici simulované hlavy stroje T. Tu bude reprezentovat následovně. Jestliže je v daném simulovaném kroku hlava T nad i-tým prvkem, pak první jednička kódu $C_A(a_i)$ bude označena tečkou (tzn. místo 1 vloží $\dot{1}$).

Pro usnadnění orientace si od sebe U oddělí jednotlivé sekce speciálními znaky. V našem případě bude začátek označen #. Kód stroje a buffer budou odděleny *. Buffer a simulační páska budou odděleny $\triangle$.
Na začátku nějakého kroku simulačního cyklu tedy může páska U vypadat například následovně:

$$\# \underbrace{1,1 \rightarrow 111{,}11; 1|1{,}11 \rightarrow ...1|}_{\text{Kód stroje}} * \underbrace{111{,}11 \rightarrow}_{\text{Buffer}} \;\triangle\; \underbrace{11|\dot{1}11|1111|1...|11|1|}_{\text{Simulační páska}} \epsilon\epsilon...$$

No a v tuto chvíli už je poměrně jasné, jak simulace bude probíhat. Na začátku každého kroku simulačního cyklu bude mít U v bufferu uložený konfigurační pár (q,a) stroje T a za ním zakódovanou jeho pracovní pásku. Při přechodu mezi jednotlivými konfiguracemi stroje T pak U bude hledat shodu tohoto konfiguračního páru s nějakým vstupem δ v kódu stroje. Podle toho, jaký tam najde výstup, tak přepíše obsah bufferu a upraví simulační pásku. Takto pokračuje dokud pro příslušný pár (q,a) tuto shodu v kódu stroje nenajde. V tom případě se T nachází v koncovém stavu. Práce U tedy není nijak složitá. Je o něco horší ji popsat algoritmicky:
Nechť $x \in \{0,1\}^*$ je vstup výpočtu stroje T, který má U za úkol simulovat. Pro ně definuji vstup stroje U jako $y(x,T) = (\alpha(T),x)$.

1. U interpretuje párovací funkci na $(\alpha(T),x)$, tzn. na vstup zavolá stroj 3.2.13 ve smyslu lemmatu 3.2.7. Potom projde pásku zleva doprava. Při tom každou první a třetí nulu nahradí ',', každou druhou nahradí '$\rightarrow$', každou čtvrtou nahradí ';' a každou pátou '|'. Ve chvíli, kdy narazí na *, tak posune řetězec začínající na pravém sousedovi $*$ o jeden prvek doprava a na uvolněné místo vloží $\triangle$.
   Nechť první znak za $\triangle$ je nyní $V \in \{0,1,\epsilon\}$. Jestliže
   - $V = 0$, pak U substituuje řetězec mezi $*$ a $\triangle$ za $11{,}1 \rightarrow$
   - $V = 1$, pak U substituuje řetězec mezi $*$ a $\triangle$ za $1{,}1 \rightarrow$
   - $V = \epsilon$, pak U substituuje řetězec mezi $*$ a $\triangle$ za $1{,}111 \rightarrow$

   Potom U projde pásku od $\triangle$ doprava a každou 1 substituuje za 1|, každou 0 substituuje za 11|. Ve chvíli, kdy narazí na první ϵ, pak jej substituuje 111|. Nakonec první 1 za $\triangle$ označí tečkou, tzn. vloží místo ní $\dot{1}$, a pokračuje krokem 2.

2. U hledá shodu řetězce obsaženého v bufferu s nějakým podřetězcem v části kódu stroje T. (viz stroj 3.2.5) Stroj shody skončí se čtecí hlavou nastavenou na nějakém prvku a

    - Pokud $a = *$, pak to znamená, že U nenašel shodu a simulovaný stroj T je v koncovém stavu. U tedy pokračuje krokem 6., ve kterém pouze poupraví pracovní pásku do finální podoby.
    - Pokud $a = 1$, pak U našel daný vstup v kódu stroje T. U Tedy vloží do bufferu výstup δ při tomto vstupu. Tedy vloží do Bufferu řetězec nacházející se mezi nejbližší → a ; zprava. Dané ; si označí jako $\underline{;}$. Potom posune pásku za $\triangle$ a na uvolněné místo vloží →. Pokračuje krokem 3.

3. U vloží kód, který je v bufferu uložen mezi , a → za kód, nad kterým je simulovaná hlava stroje T (tzn. ten na pásce stroje, který je označený tečkou), viz lemma 3.2.9. První 1 v této sekvenci opět nahradí $\dot{1}$,

4. Podle toho, jestli je v kódu stroje $\underline{;}$ součástí řetězce $\underline{;}11|$ nebo $\underline{;}1|$, stroj U posune tečku buď doprava, nebo doleva. Jestliže ji navíc posunul doleva na $\triangle$. Pak jí opět vrátí zpátky. Odoznačí ';'. Pokračuje krokem 2..

5. T nahradí prvek, který se nachází v bufferu mezi ',' a '→' řetězcem 1, který je na pásce stroje označen tečkou. Tu 1 v bufferu, která je pak označena tečkou odznačí a pokračuje krokem 2.

6. U nahradí všechny prvky mezi # a $\triangle$ symbolem ϵ. Potom postupně prochází jednotlivé kódy na pracovní pásce T a $|11|$ mění na $\epsilon\epsilon\epsilon 0$ a $|1|$ mění na $\epsilon\epsilon 1$. Ostatní $|\underbrace{11...1}_{i\geq 3}|$ změní na $\epsilon^{(i+2)}$. V tuto chvíli je na pásce výstup T proložený prázdnými znaky. Tím jsme tedy hotovi.

□

$$\#1,1 \rightarrow 111{,}11;1|1{,}11 \rightarrow ...1| * 1{,}1 \rightarrow \triangle\ 11|\dot{1}11|1111|1...|11|1|\epsilon\epsilon...$$

$$\#1,1 \rightarrow 111{,}11\underline{;}1|1{,}11 \rightarrow ...1| * \mathbf{111{,}11} \rightarrow \triangle\ 11|\dot{1}11|1111|1...|11|1|\epsilon\epsilon...$$

$$\#1,1 \rightarrow 111{,}11\underline{;}1|1{,}11 \rightarrow ...1| * 111{,}11 \rightarrow \triangle\ 11|\mathbf{\dot{1}1}|1111|1...|11|1|\epsilon\epsilon...$$

$$\#1,1 \rightarrow 111{,}11;1|1{,}11 \rightarrow ...1| * 111{,}11 \rightarrow \triangle\ 11|11|\mathbf{\dot{1}}111|1...|11|1|\epsilon\epsilon...$$

$$\#1,1 \rightarrow 111{,}11;1|1{,}11 \rightarrow ...1| * 111{,}\mathbf{1111} \rightarrow \triangle\ 11|11|\dot{1}111|1...|11|1|\epsilon\epsilon...$$

3.2.4 Řešitelná zobrazení a Problém zastavení

Celá sekce literatury se věnuje tomu, která zobrazení jsou algoritmicky řešitelná a která nikoliv. Asi nejznámějšími problémy, které se toho týkají jsou známy pod pojmy problém zastavení (=Halting problem) a problém rozhodnutí (=Entscheidungsproblem). Naší práci o Turingových strojích jsme začali problémem rozhodnutí, tak je jedině správné, abychom s ním tuto sekci i zakončili.

Definice 3.2.13. *Buď $B \subseteq \{0,1\}^*$, potom libovolné zobrazení $\psi : B \to \{0,1\}^*$ nazveme parciální funkcí. Pokud navíc existuje stroj T takový, že $D(T) = D(\psi)$ a pro každé $\omega \in D(\omega) : T(\omega) = \psi(\omega)$, tak řekneme, že ψ je (algoritmicky) řešitelná.*

V průběhu této kapitoly se budeme zabývat v podstatě výhradně řešitelností na binárních Turingových strojích. Uvědomme si však, že tato restrikce není nijak omezující. Místo širší abecedy může totiž stroj pracovat s nějakým jejím binárním kódováním. Při práci by potom vždy prostě načetl příslušný kód a pak se zachoval, tak jak by si to daný algoritmus vyžadoval. Formální argument pro zmíněnou restrikci dává následující tvrzení.

Tvrzení 3.2.16. *Nechť $T = (A,\Sigma,Q,\delta,q_h,q_s,\epsilon)$ je Turingův stroj a $\psi : A \to \{0,1\}^{\lceil \log_2(|A|) \rceil}$ je libovolné prosté zobrazení. Potom existuje binární Turingův stroj B, že pro libovolné $\alpha := a_1a_2...a_n \in \Sigma^*$, jestliže*

- *$T(\alpha) = b_1b_2...b_m$, pak*

$$B\Big(\psi(a_1)\psi(a_2)...\psi(a_n)\Big) = \psi(b_1)\psi(b_2)...\psi(b_m)$$

- *$\alpha \notin D(T)$, pak*

$$\psi(a_1)\psi(a_2)...\psi(a_n) \notin D(B)$$

Důkaz. Stroj B zkonstruujeme algoritmicky následujícím způsobem. B bude používat pouze pracovní abecedu $\{0,1\}$. B začne krokem 1., kde si bude pamatovat stav q_s a hlavu bude mít nastavenou na prvním prvku vstupu.

1. Stroj B si pamatuje stav q a na jeho pásce je řetězec $\beta\ \alpha\ \gamma$, kde platí $\ell(\alpha) = \lceil \log_2(|A|) \rceil$ a B má nastavenou hlavu na prvním znaku α. B načte α. Jestliže $q = q_h$, tak B pokračuje krokem 2. Jinak ať $\delta(q,\psi^{-1}(\alpha)) = (q,a,i)$. Potom B substituuje řetězec α za $\psi(a)$ (viz lemma 3.2.8). Pokud navíc
    (a) $i = 1$, tak B udělá $2\lceil \log_2(|A|) \rceil$ kroků od prvního znaku α doprava a zapamatuje si stav $\tilde{q}$
    (b) $i = -1$, tak udělá $\lceil \log_2(|A|) \rceil$ kroků od prvního znaku α doleva a zapamatuje si stav $\tilde{q}$
2. B projde konfigurační řetězec zleva doprva. Vždy načte prvních $\lceil \log_2(|A|) \rceil$ neprojitých znaků a každé $\psi(a) \notin \{\psi(1),\psi(0)\}$ při tom substituuje za $\epsilon^{\lceil \log_2(|A|) \rceil}$ (viz lemma 3.2.8). Jakmile při tom narazí na první ϵ, tak skončí práci.

□

Roku 1928 jeden z nejvlivnějších matematiků své doby, David Hilbert, formuloval tři dodatečné problémy k jeho slavným 23 problémům pro nadcházející století. Mezi nimi byl i problém rozhodnutí. Jednalo se o součást takzvaného Hilbertova programu. Toto hnutí koncentrované zejména na Gotingenské universitě, bylo zažehnuto Hilbertovým snem k vytvoření syntaktického systému, který by byl schopný reprezentovat argumentaci v rámci axiomatických teorií v čistě symbolické podobě. Z tohoto myšlenkového proudu do velké míry vychází současná dominantní filosofie matematiky, známá jako formalismus. (Bezhanishvili a Moss, 2019)
Myšlenka axiomatizace matematiky nebyla ničím novým, objevovala se již ve starém Řecku. V čem byla Hilbertova vize unikátní, byla formální definice důkazu. Zatímco v Řecké filosofii pojmy důkaz a axiom měly co dočinění s pravdou, tak v Hilbertově teorii se jednalo čistě o syntaktické objekty. (Wolf, 2005)
V tomto textu (axiomatickou) teorií míníme

1. nějakou logiku (někdy také jazyk), určující povolené symboly (funkční, relační, proměnné,...) a pravidla, jak dané symboly spojovat do platných logických formulí,

2. společně s vlastními axiomy dané logiky (vybranými formulemi v rámci příslušné logiky).

Formální detaily ke všem definicím (i pro zbytek kapitoly) jsou obsaženy například v (Švejdar, 2002, kapitoly 3.1-3.2).
Hilbertova myšlenka důkazu v matematické teorii byla ve skutečnosti celkem jednoduchá. Hilbert formuloval dnes již poměrně známé axiomy logiky a pravidla inference, tzn. pravidla, která reprezentovala, co tvoří validní krok důkazu. Tvrzení bylo dokazatelné z axiomů dané teorie, jestliže existovala sekvence formulí v dané logice, kde každá z těchto formulí byla buď nějaký z příslušných axiomů (ať už vlastních, nebo axiomů logiky) anebo vznikla z přechozích tvrzení na základě aplikování pravidla inference.

Značení. *Nechť T je teorie (v logice L). Fakt, že formule φ v logice L je dokazatelná v T budeme značit*

$$T \vdash \varphi$$

Příklad 3.2.7. *Všechna pravidla inference zde ukazovat nebudeme. V zásadě se jedná o jednoduché vztahy jako modus ponens (pokud platí f a $f \to g$, potom platí g)*

$$\begin{gathered} f \\ f \to g \\ \because \\ g \end{gathered}$$

Hlavní snaha Hilbertova programu se potom otáčela kolem toho, jaká obecná fakta o matematice jdou z této definice vyvodit. Platí například, že každé tvrzení, nebo jeho negace jsou v teorii dokazatelné z axiomů? Lze demonstrovat, že je daná teorie bezesporná? Existuje algoritmus, který by ukázal, jestli je tvrzení v dané teorii dokazatelné z axiomů? Poslední z těchto požadavků je právě problém rozhodnutí:

> *„Nyní jsme narazili na fundamentální problém: Je možné rozhodnout, jestli je dané tvrzení vztahující se k nějaké oblasti vědění důsledkem axiomů?"*
>
> -Lze nalézt např. v knize (Hilbert a Ackermann, 1999, str. 108)

Pokud by tyto podmínky byly splněny, tak Hilbert předpokládal, že by jejich důkaz poskytnul dostatečný filosofický podklad pro matematickou argumentaci. První dva cíle tohoto programu se rozpadly v prach ve chvíli, kdy Gödel dokázal dvě věty o neúplnosti. Po roce 1936 tedy zůstal nezodpovězen pouze problém rozhodnutí. Turing a Church nezávisle na sobě dokázali, že takový algoritmus nalézt nelze. Turing v podstatě daný problém redukoval na algoritmicky neřešitelné otázky o jeho strojích. Jednou z takových otázek je již zmíněný problém zastavení:

Problém zastavení volá po algoritmu, který by na vstupu obdržel Turingův stroj, vstup do tohoto stroje a rozhodnul by, jestli se příslušný stroj na tomto vstupu někdy zastaví, nebo ne. Jak Turing ukázal, tak tento problém nemá řešení. Myšlenka jeho důkazu je nesmírně elegantní.

Než se do problému zastavení ovšem pustíme, tak je třeba odůvodnit některé základní poznatky o řešitelnosti problémů spojených s Turingovými stroji.

Poznámka 3.2.17. *V následujícím textu budeme dokazovat, že některá fakta o Turingových strojích nejsou algoritmicky zjistitelná. Abychom taková tvrzení mohli naformulovat, tak ovšem budeme muset ztotožnit každý Turingův stroj s nějakým binárním řetězcem. Od takového ztotožnění vyžadujeme nějaké základní vlastnosti spojené s interpretovatelností vstupu. Jmenovitě potřebujeme, aby byl okolo něj zkonstruovatelný stroj, který by na něm byl schopný vypsat:*

- *Velikost množiny vnitřních stavů*
- *Velikost pracovní abecedy*
- *Výstup transformční funkce při libovolně zvoleném vstupu*

$$\vdots$$

Jinými slovy od onoho ztotožnění očekáváme, že obsahuje všechnu informaci danou v definici příslušného stroje v jakési algoritmicky interpretovatelné podobě. Myšlenka je taková, že kdyby měl být nějaký problém algoritmicky řešitelný, tak bychom intuitivně kolem takového vstupu měli být schopni zkonstruovat stroj, který jej bude při dané interpretaci řešit. Není těžké si rozmyslet, že poměrně dobrou volbou je již definované číslo $\alpha(T)$.

Značení. *Na základě předchozí diskuze pro binární Turingův stroj* T *označme*

$$(T,x) := (\alpha(T),x)$$
$$(T) := \alpha(T)$$

Poznámka 3.2.18. *Existenci universálního stroje, kterou jsme dokázali ve větě 3.2.15, můžeme podle nově nabytého značení přeformulovat následovně: Existuje algoritmicky řešitelné parciální zobrazení* ψ, *že* $\psi(T,\omega) = T(\omega)$ *pro každé* $\omega \in D(\psi)$ *a které splňuje, že jestliže* $\omega \notin D(T)$, *potom* $(T,\omega) \notin D(\psi)$.

K důkazu nerozhodnutelnosti problému zastavení potřebujeme následující lemma.

Lemma 3.2.19. *Existuje binární Turingův stroj T, který na každém vstupu $x \in \{0,1\}^*$ dá výstupní konfiguraci s řetězcem*

$$(x,x) = 1^{\ell(x)}0xx$$

Důkaz. Stroj $T = (A,\Sigma,Q,\delta,q_h,q_s,\epsilon)$ bude pracovat následovně:

1. T posune vstup dvakrát doprava a na uvolněné místo vloží řetězec '$*\#$' (stroj 3.2.3). Potom místo prvního ϵ na pásce vloží $\triangle$ a za něj zkopíruje x. (například substituční stroj 3.2.9)
2. Nyní je na pásce řetězec $* \ \alpha \ \# \ \beta \ \triangle \ x$, kde $\alpha, \beta \in A^*$. Nechť a je první neoznačený prvek za $\#$.
   (a) $a \neq \triangle$, tak T označí a. Potom posune celý řetězec začínající na pravém sousedovi $*$ doprava a na uvolněné místo vloží 1.
   (b) $a = \triangle$, tak T odznačí všechny prvky β, nahradí $\#$ symbolem 0. Posléze posune řetězec začínající na pravém sousedovi $\triangle$ o jednu pozici doleva. Potom posune celý řetězec začínající na pravém sousedovi * o jeden prvek doleva (viz lemma (3.2.4)) a skončí práci.

□

Věta 3.2.20. *(Problém zastavení) Mějme libovolné parciální zobrazení Halt splňující*

$$Halt(T,x) = \begin{cases} 1, & \textit{pokud } T \textit{ skončí práci na } x \\ 0, & \textit{jinak} \end{cases}$$

pro každý binární Turingův stroj T a $x \in \{0,1\}^*$. Potom zobrazení Halt není řešitelné.

Důkaz. Toto tvrzení ukážeme sporem. Nechť tedy existuje stroj H řešící $Halt$. Neboli ať stroj H splňuje

$$H(\alpha(T),x) = \begin{cases} 1, & \text{pokud } T \text{ skončí práci na } x \\ 0, & \text{jinak} \end{cases}$$

Ukážeme, že z existence H plyne existence stroje M, který se na vstupu $\alpha(M)$ nemůže ani zastavit ani být v nekonečném cyklu. M bude používat H jako knihovní funkci ve smyslu tvrzení 3.2.7. Definuji jeho práci na vstupu $x \in \{0,1\}^*$ následovně:

1. M převede vstup x do tvaru (x,x) (lemma 3.2.19).
2. Spustí stroj H ve smyslu tvrzení 3.2.7.
3. Na pásce je nyní jediné $a \in \{0,1\}$, M jej najde.
4. Pokud $a = 1$, tak se M zacyklí (viz stroj 3.2.1) a pokud $a = 0$, tak M skončí práci.

Tedy stroj M je definovaný přesně tak, aby na vstupu $\alpha(T)$ skončil práci právě tehdy, když na něm T práci neskončí. Uvažujme nyní vstup $\gamma := (\alpha(M),\alpha(M))$ do H. Protože M je binární Turingův stroj a $\alpha(M) \in \{0,1\}^*$ je vstup do M jako každý jiný, tak H jistě na γ skončí práci. Výstup $v = H(\gamma) \in \{0,1\}$ je tedy dobře definovaný. Kdyby nyní v bylo 1, pak by M skončilo práci na $\alpha(M)$. Jenomže definice M nám v tomto případě určuje, že M neskončí práci na $\alpha(M)$. To je očividný spor. Případ kdy $v = 0$ vede ke sporu analogicky. □

Zmíněné tvrzení samo o sobě nese poměrně důležité důsledky pro teorii programování. U některých algoritmů jsme okamžitě schopni říct, že se na nějakém vstupu daný program po určitém počtu kroků zastaví. U některých, že se zase za některých podmínek ocitne v nekonečném cyklu. Ať už ovšem vymyslíme libovolný program, který bude mít za úkol tuto skutečnost testovat, tak budou existovat dvojice algoritmu a vstupu, na nichž náš postup selže. Ukazuje se dokonce, že tento problém platí i v silnější podobě. Problém zastavení totiž není řešitelný ani ve chvíli, kdy zafixujeme prázdný vstup. O této modifikaci budeme mluvit jako o prázdném problému zastavení.

Lemma 3.2.21. *Buďte S_1 a S_2 stroje s pracovní abecedou A, potom existuje stroj T, který na libovolném vstup*

$$W \; \alpha \; \beta \; Y \; \gamma \; Z$$

kde $W,X,Y,Z \in A$ nejsou obsaženy v řetězcích α,β,γ^*, jestliže

- *$\ell(\alpha) < \ell(\gamma)$, tak spustí na příslušném vstupu stroj S_1*
- *$\ell(\alpha) \geq \ell(\gamma)$, tak spustí na příslušném vstupu stroj S_2*

Důkaz.

1. Nechť b je první neoznačený prvek za Y. Pokud
    - $b = Z$, tak každý prvek na pásce odznačí a spustí stroj S_2
    - $b \neq Z$, tak nechť a je první neoznačený prvek za W. Pokud potom
        - $a = X$, pak odznačí všechny prvky na pásce a spustí stroj S_1
        - $a \neq X$, tak označí a i b a pokračuje krokem 1.

□

Tvrzení 3.2.22. *Buď $Halt_\varnothing$ libovolné parciální zobrazení splňující*

$$Halt_\varnothing(T) = \begin{cases} 1, & \text{pokud } T \text{ skončí práci na } \varnothing \\ 0, & \text{jinak} \end{cases}$$

pro každý binární Turingův stroj T. Potom $Halt_\varnothing$ není řešitelné.

Důkaz. Toto tvrzení ukážeme sporem. Nechť tedy $Halt_\varnothing$ je řešitelné. Označme potom S stroj, který řeší prázdný problém zastavení. Chceme ukázat, že odsud existuje stroj H, který bude řešit $Halt$. H bude používat S jako knihovní funkci. Mějme tedy libovolný stroj T a $\omega \in \{0,1\}^*$. Pro ně označme T_ω stroj, který nejprve přepíše prvních $\ell(\omega)$ znaků vstupu na ω, pak posune hlavu na začátek pásky a pokračuje ve výpočtu jako T. Tedy T_ω v podstatě pracuje na prázdném vstupu, jako T na ω. Práci H pak definujme na vstupu $(\alpha(T),\omega)$ následovně:

1. H převede $(\alpha(T),\omega)$ do tvaru $\#\alpha(T) * \omega$ (viz stroj 3.2.13)
2. H převede $\#\alpha(T) * \omega$ na $\alpha(T_\omega)$.
3. H pustí na vstup S. (viz stroj 3.2.7)

Zbývá si rozmyslet, že krok 2., tzn. konverze $\#\alpha(T) * \omega \to \alpha(T_\omega)$, je opravdu korektní. Intuitivně je jasné, co je potřeba udělat. Pokud $T = (A,\Sigma,Q,\delta,q_h,q_s,\epsilon)$, kde $A = \{a_1,a_2,...,a_n\}$ a $\omega = \omega_1\omega_2...\omega_v$, tak musíme před daný řetězec vložit postupně přes $i = 1,2,...,n$ a $j = 1,2,...,m$ všechny řetězce

$$1^{(i)}\ 0\ 1^{(j)}\ 0\ 1^{(i+1)}\ 0\ C_A(\omega_i)\ 1\ 0$$

, tedy část kódu zodpovědnou za výpis ω a potom v každém řetězci $C^T(q,a,\tilde{q},\tilde{a},i)$ původního $\alpha(T)$ všechny $C_Q(\tilde{q})$ a $C_Q(q)$ rozšířit o v jedniček (tzn přeindexovat zbylé stavy). Není těžké si rozmyslet, že takto upravený řetězec je právě $\alpha(T_\omega)$. Detailně to lze provést například takto:

1. Na pásce je nyní $\#\alpha(T) * \omega$. Nechť a je první prvek za $*$. Jestliže
   - $a \neq \epsilon$, tak T substituuje prázdný řetězec mezi $\#$ a první 1 za

     $$1\ 0\ 1\ 0\ 1\ 1\ 0\ C_A(a)\ 0\ 1\ \triangle$$

     (viz 3.2.8). Pokračuje krokem 2.
   - $a = \epsilon$, tak skončí práci.
2. Na pásce je nyní

   $$\text{'}\#\ 1^{(i_1)}\ 0\ 1^{(i_2)}\ 0\ 1^{(i_3)}\ 0\ 1^{(i_4)}\ 0\ 1^{(i_5)}\ 0\ \alpha\ \triangle\ \beta\ 0\ 1^{(n_1)}\ 0$$
   $$1^{(n_2)}\ 0\ 1^{(n_3)}\ 0\ 1^{(n_4)}\ 0\ 1^{(n_5)} * \bar{\omega}\text{'}$$

   pro nějaká $i_k,n_k \in \mathbb{N}_0$, $\alpha,\beta \in \{0,1\}^*,\bar{\omega} \in \{0,1,\dot{0},\dot{1}\}$. T porovná i_2,n_2 (viz lemma 3.2.21). Jestliže
   - $i_2 < n_2$, tak T substituuje (stroj 3.2.9) prázdný řetězec mezi $\#$ a 1 řetězcem

     $$1^{(i_1)}\ 0\ 1^{(i_2)}\ 0\ 1^{(i_3)}\ 0\ 1^{(i_4)}\ 0\ 1^{(i_5)}\ 0$$

     Potom posune řetězec počínající $1^{(i_2)}$ tohoto řetězce a na uvolněné místo vloží 1.
   - $i_2 = n_2$, tak T porovná i_1 a n_1 (viz lemma 3.2.21). Nechť a je první neoznačený prvek za $*$. Pokud
     - $a \neq \epsilon$, pak T substituuje (viz 3.2.9) prázdný řetězec mezi $*$ a 1 za

       $$1^{(i_1)}\ 0\ 1^{(i_2)}\ 0\ 1^{(i_3)}\ 0\ 1^{(i_4)}\ 0\ 1^{(i_5)}$$

       Potom posune řetězec začínající na pravém sousedovi * a na uvolněné místo vloží 1. Potom řetězec nově vložené $1^{(i_2)}$ (blíže k $*$) substituuje (viz 3.2.8) za $C_A(a) \in \{1,11\}$. Označí a a pokračuje krokem 3.

    - $a = \epsilon$, tak S nahradí každý prvek řetězce '$* \,\bar{\omega}$' znakem ϵ a pokračuje krokem 3.

3. Na pásce je nyní

$$\# \ \alpha \ 0 \ \zeta \ 0 \ 1^{(i_2)} \ 0 \ 1^{(i_3)} \ 0 \ 1^{(i_4)} \ 0 \ 1^{(i_5)} \ 0 \ \triangle \ \beta$$

   pro nějaká $\zeta \in \{1,\dot{1}\}^*$, $\alpha,\beta \in \{0,1\}^*$. Máme dvě možnosti. Buď

   - existuje prvek a řetězce ζ, že $a = 1$. Potom S označí a. Dále víme, že řetězec β se skládá výhradně z podřetězců $1^{(n)}0$ pro $n \in \mathbb{N}$. S tedy projde β zleva doprava a vždy řetězec začínající na každém druhém a čtvrtém takovém podřetězci (modulo 5) posune doprava a na uvolněné místo vloží 1. Pokračuje krokem 3.
   - každý prvek ζ je roven $\dot{1}$. Proto S odznačí všechny prvky na pásce a řetězec začínající na pravém sousedovi $\triangle$ posune o jeden prvek doleva. Posléze skončí práci.

□

Na začátku této kapitoly jsme mluvili o Hilbertově škole a o syntaktizaci matematiky. Přesto, že tyto myšlenky byly ve své době nesmírně důležité, tak ani zdaleka nepopisují celý příběh logiky počátku 20. století. K důkazu některých fundamentálních důsledků jako byla Skolemova věta a Gödelova věta o úplnosti, bylo třeba opustit myšlenku syntaktického objektu a zamyslet se nad jeho interpretací. To souvisí s oddělením syntaxe výrazu od jeho sémantiky. Pro ilustraci uvažujme: (Avigad)

$$\varphi(x) := x^3 + \exp(\ln(x^2))$$

Je φ polynom? Naším prvním impulzem by pravděpodobně bylo říct ano, protože „$\varphi(x) = x^3 + x^2$“. Ovšem jako formální výraz, tedy posloupnost symbolů, $\varphi(x)$ polynomem není. To, co je v tomto případě zavádějící je, že výrazy $\varphi(x)$ a $x^3 + x^2$ definují stejné reálné zobrazení. Tedy interpretujeme-li dané výrazy jako funkce se všemi operacemi, které k tomu náleží: jako je sčítání a násobení funkcí atp., tak se při této intepretaci dané výrazy shodují. Dnes by se v případě, kdy by to hrálo roli mluvilo o syntaktickém zápisu $x^3 + x^2$ jako o formálním polynomu a o zobrazeních, která jdou vyjádřit nějakým polynomiální předpisem (např $\varphi(x)$), jako o polynomiálním zobrazení. Rozlišení mezi matematickým objektem a jeho syntaktickým zápisem bylo dlouho dobu opomíjeno. Matematická logika pracuje s logickými formulemi jako s matematickými objekty. V jejím případě je tento problém ještě o něco markantnější. Uvažujme například následující formuli v nějakém jazyce

$$\forall x : x + y = y + x$$

To, jak nad daným tvrzením budeme přemýšlet a to jestli jej budeme považovat za pravdivé, silně záleží jak na jeho syntaxi (na tom, že $+$ je binární funkční symbol, na tom jestli x,y jsou volné či vázané proměnné,...), tak na možné interpretaci, kterou tomuto výrazu můžeme na základě této syntaxe přisoudit (jsou x,y vektory, nebo čísla? co značí operace $+$?).
O takové interpretaci jazyka L se často mluví jako o struktuře pro L. Formálně

je struktura nějaké logiky dvojice nosné množiny a zobrazení logických relačních/funkčních symbolů daného jazyka do příslušných relací a funkcí definovaných nad onou nosnou množinou zachovávající aritu a relační znak '='. Pokud jsou axiomy toerie T splněny (=pravdivé) v nějaké struktuře $\mathbf{D}$ (při interpretaci daných symbolů) na libovolném ohodnocení jejích proměnných, tak se říká, že daná struktura je modelem teorie T. Formuli f, která je pravdivá v každém modelu příslušné teorie, pak nazýváme validní (v rámci teorie T). Jinými slovy formule je validní právě tehdy, když je pravdivá, nehledě na interpretaci, kterou nám zvolené axiomy povolují. Tato definice se někdy nazývá Tarského definicí pravdy. Fakt, že ji f v teorii T splňuje značíme $T \models f$. Na základě zmíněné diskuze můžeme formulovat Gödelovu větu o úplnosti. (Technické detaily včetně zmíněných definic opět v (Švejdar, 2002))

Věta 3.2.23. *(Gödel) Je-li T teorie a f formule v T, pak*

$$T \vdash f \text{ právě tehdy, když } T \models f$$

To nám dává způsob, jak přeformulovat problém rozhodnutí:

> *Mějme libovolnou toerii prvního řádu. Je možné nějak obecně algoritmicky rozhodnout o tom, zda-li je libovolně zvolená formule v dané teorii validní?*

Centrální myšlenkou důkazu neřešitelnosti problému rozhodnutí bude prázdný problém zastavení. Pokusíme se vytvořit teorii prvního řádu a v ní pro každý stroj M tvrzení f_M, kde f_M bude validní právě tehdy, když M nedefinuje výstup na prázdném vstupu. Protože tento fakt nelze algoritmicky rozhodnout, pak nejde rozhodnout ani o validnosti tvrzení v dané teorii. Jinými slovy problém rozhodnutí zredukujeme na prázdný problém zastavení.

Při vytváření příslušné logiky máme na mysli jakýsi zamýšlený model. Představme si, že by nosná množina daného modelu byla množina políček pásky ve všech možných krocích (tzn. dvojice čas, políčko). Definice dané logiky $\mathcal{L}$ by obsahovala jednu konstantu pocatek, jejíž interpretace by byla první políčko na začátku prvního kroku výpočtu stroje M na prázdné pásce. Potom bychom v dané logice měli unární relace: vstup(x) - pravdivý v daném modelu, pokud se jedná o prvek vstupní pásky (na libovolné pozici), kraj(x) - pravdivý pokud se jedná o kraj pásky (v libovolném kroku), $H(x)$ - pravdivý pokud je v daném kroku hlava M nad danou pozicí a Q_l,S_k, které značí, jestli je v daný čas na daném místě symbol s_k resp. jestli je stroj ve stavu q_l. Funkční symboly $L(x)$, $P(x)$ by pak dávaly levého, resp. pravého souseda políčka v daném čase a krok(x) tu samou pozici na pásce v následujícím kroku. Pomocí těchto funkčních/relačních symbolů s touto interpretací na mysli vytvoříme axiomy $f_1 - f_{16}$ teorie $\mathcal{T}_M$, které mají za úkol popsat práci stroje M. Výsledné tvrzení f_M pak bude říkat, že stroj M neskončí práci na prázdném vstupu. Protože libovolný model této teorie musí nutně splňovat všechny axiomy $f_1 - f_{16}$ (na libovolném ohodnocení), tak díky tomu bude tvrzení f_M validní právě tehdy, když daný stroj nedefinuje na prázdném vstupu výstup.

Značení. *V následujících dvou větách budeme používat korespondenci mezi znaky s_i a relacemi S_i. Abychom usnadnili značení tak ztotožníme $s_\epsilon := \epsilon$.*

Definice 3.2.14. *Definuji logiku prvního řádu $\mathcal{L}$ následovně. $\mathcal{L}$ kromě povinných symbolů obsahuje:*

- *nulární funkční symbol: pocatek*
- *unární relační symboly: vstup, kraj, H, Q_s, Q_h, Q_1, Q_2,...,*
  S_ϵ, S_1, S_2,...
- *unární funkční symboly: krok, L, P*

Dále uvažujme stroj $M = (A,\Sigma,Q,\delta,q_h,q_s,\epsilon)$, kde $A = \{s_1,s_2,...,s_n,\epsilon\}$, $Q = \{q_s,q_h,q_1,...,q_m\}$. Pro něj v rámci logiky $\mathcal{L}$ zkonstruujeme následující formule:

1. *Počátek (v uvažovaném modelu první pozice vstupu) je součástí vstupu*

$$vstup(pocatek)$$

2. *Každý prvek vstupu je buď počátek, nebo pravý soused nějakého prvku vstupu, tzn. vstup tvoří „polopřímku“ (či více „polopřímek“)*

$$(\forall x)\Big(vstup(x) \leftrightarrow \Big(\ (pocatek = x) \vee (\exists y)(vstup(y) \wedge P(y) = x)\Big)\Big)$$

3. *Funkce krok( ) je prostá, tedy políčko je s inkrementací času určeno jednznačně:*

$$(\forall x,y)\Big(\Big(krok(x) = krok(y)\Big) \rightarrow x = y\Big)$$

4. *Pokud se podíváme v následujícím kroku na pravého souseda nějakého políčka, tak je to to samé jako bychom se koukli na pravého souseda (příslušného políčka) v následujícím kroku.*

$$(\forall x)\Big(krok\Big(P(x)\Big) = P\Big(krok(x)\Big)\Big)$$

5. *Analogie 4. pro L*

$$(\forall x)\Big(krok\Big(L(x)\Big) = L\Big(krok(x)\Big)\Big)$$

6. *Až na případ, kdy je políčko na kraji, jsou P a L inverzní*

$$(\forall x)\Big(\Big(P(L(x)) = x\Big) \wedge \Big(\neg kraj(x) \rightarrow L(P(x)) = x\Big)\Big)$$

7. *Políčko je něčím pravým sousedem právě tehdy, když není na kraji:*

$$(\forall x)\Big(\neg kraj(x) \leftrightarrow (\exists y)(P(y) = x)\Big)$$

8. *Pokud má stroj posunout hlavu doleva z kraje, tak ji z definice ponechá na místě. Jinak posunutí doleva a doprava neponechají hlavu na stejném místě.*

$$(\forall x)\Big(\Big(L(x) = x \leftrightarrow kraj(x)\Big) \wedge \neg(P(x) = x)\Big)$$

9. *Počátek je na kraji pásky a každý prvek kraje je na kraji i o krok dál*

$$kraj(pocatek) \wedge \Big((\forall x)\Big(kraj(x) \rightarrow (kraj(krok(x)))\Big)\Big)$$

10. *Na každé pozici je v každém kroku pouze jeden prvek*

$$(\forall x)\Big(\bigvee_{s_i \in A}\Big(S_i(x) \wedge \bigwedge_{s_j \in A, s_j \neq s_i} \neg S_j(x)\Big)\Big)$$

11. *V každém čase (a na každé pozici) jsme pouze v jednom stavu*

$$(\forall x)\Big(\bigvee_{q_i \in Q}\Big(Q_i(x) \wedge \bigwedge_{q_j \in Q, q_j \neq q_i} \neg Q_j(x)\Big)\Big)$$

12. *Na vstupu jsou prázdné znaky, stroj je v počátečním stavu s hlavou nastavenou na první pozici.*

$$(\forall x)\Big(\Big(vstup(x) \rightarrow (S_\epsilon \wedge Q_s(x) \wedge (H(X) \leftrightarrow x = pocatek)\Big)\Big)$$

13. *S inkrementací času se nezmění prvek zapsaný na políčkách, nad kterými není nastavena hlava stroje:*

$$(\forall x)\Big(\bigwedge_{s_i \in A}\Big(\neg H(x) \wedge S_i(x)\Big) \rightarrow S_i(krok(x))\Big)$$

14. *Nechť navíc $\delta(q_i, s_j) = (q_d, s_b, -1)$, potom se s aplikací krok( ) změní hodnoty Q_d, S_b a H tak, aby odpovídali pásce stroje v následujícím kroku (s aplikací L se změní i hodnota H)*

$$(\forall x)\Big[\bigwedge_{q_i \in Q, s_j \in A}\Big(H(x) \wedge Q_i(x) \wedge S_j(x)\Big) \rightarrow$$
$$\Big(H(krok(L(x))) \wedge Q_d(krok(L(x))) \wedge S_b(krok(x))\Big)\Big]$$

15. *Nechť navíc $\delta(q_i, s_j) = (q_d, s_b, 1)$, potom se s aplikací krok( ) změní hodnoty Q_d, S_b tak aby odpovídali pásce stroje v následujícím kroku (s aplikací P se změní i hodnota H)*

$$(\forall x)\Big[\bigwedge_{q_i \in Q, s_j \in A}\Big(H(x) \wedge Q_i(x) \wedge S_j(x)\Big) \rightarrow$$
$$\Big(H(krok(P(x))) \wedge Q_d(krok(P(x))) \wedge S_b(krok(x)\Big)\Big]$$

16. *Do každé konfigurace, vyjma té počáteční, se musel stroj nějak dostat na základě předchozích konfigurací a transformační funkce*

$$(\forall x)\Big[\bigwedge_{q_i \in Q, s_j \in A}\Big(H(x) \wedge Q_i(x) \wedge S_j(x)\Big) \rightarrow$$
$$\Big((x = pocatek)$$
$$\vee (\exists y)\Big((krok(L(y)) = x) \wedge \bigvee_{\delta(q_a, s_b) = (q_i, s_j, -1)} (H(y) \wedge Q_a(y) \wedge S_b(y))\Big)$$
$$\vee (\exists y)\Big((krok(P(y)) = x) \wedge \bigvee_{\delta(q_a, s_b) = (q_i, s_j, 1)} (H(y) \wedge Q_a(y) \wedge S_b(y))\Big)\Big)\Big]$$

17. *Na každém prvku, nad kterým je nastavena hlava platí, že není v koncovém stavu (tzn. M nikdy neskončí práci):*

$$(\forall x)\Big(H(x) \to \neg Q_h(x)\Big)$$

Označme tvrzení z bodu $i = 1,2,...,17$ jako f_i, potom v logice $\mathcal{L}$ definuji teorii $\mathcal{T}_M$ s vlastními axiomy $T_M := \{f_1,f_2,...,f_{16}\}$ a tvrzení $f_M := f_{17}$.

V následujícím textu budeme ukazovat, co je možné z těchto axiomů usuzovat o obecném modelu naší vytvořené teorie. Nejprve je však možná dobré si uvědomit, že modely této teorie se mohou chovat všelijak. Nesmíme se tedy moc spoléhat na podobu obecného modelu s výpočtem Turingova stroje. Takový model například totiž nemusí být vůbec spočetný. Obecně v něm mohou vznikat jakési podstruktury izolované našimi operacemi atp..

Značení. *Pro usnadnění značení položme $(t,m) := krok^{t-1}(P^{m-1}(pocatek))$ pro všechna $m \in \mathbb{N}$, kde $P^m(x) = \underbrace{P(P(P(....P(P}_{m}(x)))..))$ (podobně pro $krok^t$)*

Je však dobré si povšimnout, že v každé struktuře, která naše axiomy $f_1 - f_{16}$ splňuje, existuje jakási podstruktura uzavřená na operace $krok(\ )$, $R(\ )$, $L(\ )$. Vygenerujeme ji z konstanty *pocatek* jako $\bigcup\limits_{m,t} P^m(krok^t(pocatek))$. O takto generovaných množinách se někdy v literatuře mluví jako o minimální struktuře. Za chvíli ukážeme, že tímto způsobem vytvořená struktura se již v jistém smyslu chová analogicky s výpočtem uvažovaného stroje M.
A co, že tím analogickým chováním myslíme? V první řadě se přesvědčíme, že $S_i(t,n)$ bude platit právě tehdy, když v t-tém kroku výpočtu bude mít stroj M na n-tém prvku pásky prvek $s_i \in A$. Podobně $H(t,n)$ bude platit právě tehdy, když v daném kroku bude na příslušném prvku n hlava stroje. A nakonec aktuální stav stroje půjde vyčíst z platnosti relace $Q_j(t,m_t)$, kde m_t je pozice hlavy stroje v t-tém kroku. Zmiňovaná fakta dokazuje následující lemma.

Lemma 3.2.24. *Nechť $M = (A,\Sigma,Q,\delta,q_h,q_s,\epsilon)$ je binární Tuingův stroj a uvažujme nějaký model $\boldsymbol{B}$ teorie $\mathcal{T}_M$. Ať má navíc M v t-tém kroku, $t \in \mathbb{N}$ výpočtu na prázdném vstupu hlavu nastavenou na m_t-ém prvku pásky pro nějaké $m_t \in \mathbb{N}$ a je ve stavu $q_j \in Q$. Pak jsou na struktuře $\boldsymbol{B}$ splněny relace*

$$H(t,m_t) \text{ a } Q_j(t,m_t)$$

Pokud se dále na n-té pozici pásky, $n \in \mathbb{N}$ v t-tém kroku nachází znak $s_{i_n} \in A$, pak nastává

$$S_{i_n}(t,n).$$

Pro všechna $q_j \neq q_k \in Q$, $s_{i_n} \neq s_v \in A$ navíc dostaneme

$$\neg Q_k(t,m_t) \text{ a } \neg S_v(t,n)$$

a pro $n \neq m_t, n \in \mathbb{N}$ k tomu

$$\neg H(t,n).$$

Důkaz.

1. Buď $t = 1$. Všimněme si, že z f_1 a f_2 plyne vstup$(1{,}k)$ pro každé $k \in \mathbb{N}$. Tedy z f_{12} platí relace $\boldsymbol{S_\epsilon(1{,}k)}$ a $\boldsymbol{Q_s(1{,}k)}$. Pokud ke všemu $k \neq 1$, tak z f_{12} dostaneme $\boldsymbol{\neg H(1{,}k)}$ a ve zbylém případě $k = 1$: $\boldsymbol{H(1{,}1)}$. Z f_{10} navíc víme, že pro $\epsilon \neq s_i \in A$ a pro libovolné $k \in \mathbb{N}$ platí $\boldsymbol{\neg S_i(1{,}k)}$. Formule f_{11} k tomu říká, že pro $q_s \neq q_j \in Q$ platí $\boldsymbol{\neg Q_j(1{,}1)}$. To vše odpovídá počáteční konfiguraci stroje M a tedy i tomu, co tvrdíme.

2. Buď $t \in \mathbb{N}, t > 1$. Z indukčního předpokladu existují jednoznačně určená $s_i \in A, q_j \in Q$, že $S_i(t{,}m_t)$ a $Q_j(t{,}m_t)$. Dále si uvědomme, že jelikož M na prázdném vstupu nedefinuje výstup, tak jistě udělá alespoň $t + 1$ kroků. Odsud δ definuje pro pár $(q_i{,}s_j)$ odpovídajícímu t-té konfiguraci stroje M (na prázdném vstupu) výstup. Proto nastává předpoklad některé z implikací v f_{14} a f_{15}. Bez újmy na obecnosti předpokládejme, že platí nějaký z předpokladů jedné z implikací f_{14}, tedy že $\delta(q_i{,}s_j) = (q_d{,}s_b, -1)$ pro nějaká $q_d \in Q$, $s_b \in A$. Případ f_{15} se ukáže analogicky. Připomeňme, že z indukčního předpokladu platí $H(t{,}m_t)$. Z f_{14} tedy dostaneme $H(\text{krok}(L(t{,}m_t)))$, $Q_d(\text{krok}(L(t{,}m_t)))$ a $S_b(\text{krok}(t{,}m_t)) = \boldsymbol{S_b(t{+}1{,}m_t)}$. Uvědomme si navíc, že z f_9 a f_7 jsou ta $x \in \bigcup\limits_{k,m} (k{,}m)$ splňující $\mathit{kraj}(x)$ právě tvaru $\bigcup\limits_{k} (k{,}1)$. Odsud

   $$\begin{aligned} H(\text{krok}(L(t{,}m_t))) &= H(\text{krok}(L(\text{krok}^{t-1}(P^{m_t-1}(\text{pocatek}))))) \\ &\overset{f_5}{=} H(\text{krok}(\text{krok}^{t-1}(L(P^{m_t-1}(\text{pocatek}))))) \\ &\overset{f_6,f_8}{=} H(\text{krok}^t(P^{\max\{m_t-2,0\}}(\text{pocatek}))) \end{aligned}$$

   Z platnosti předpokladu f_{14} víme, že $\max\{m_t - 1{,}1\} = m_{t+1}$. V návaznosti na tento řetězec rovností tedy dostáváme platnost $\boldsymbol{H(t + 1{,}m_{t+1})}$. Analogicky odsud plyne a $\boldsymbol{Q_b(t + 1{,}m_{t+1})}$.
   Pro ta políčka, nad kterými není v t-tém čase nastavena hlava, tzn. ta jejichž pozice spadá pod nějaké $n \in \mathbb{N}$, $n \neq m_t$ z I.P. existuje právě jedno $s_{i_n} \in A$, že nastane $S_{i_n}(t{,}n)$. Ale my z indukčního předpokladu víme, že platí $\boldsymbol{\neg H(t{,}n)}$. Tedy z f_{13} máme $\boldsymbol{S_{i_n}(t + 1{,}n)}$. Formule f_{10} k tomu říká, že platí $\boldsymbol{\neg S_v(t + 1{,}n)}$ pro libovolné $s_{i_n} \neq s_v \in A$. Tzn. dané relace $\boldsymbol{S_{i_n}}$ jsou pro všechna políčka určeny jednoznačně. Nakonec si uvědomme, že z f_{16} pro každé $n \neq m_{t+1}$ dostaneme $\boldsymbol{\neg H(t + 1{,}n)}$. Z f_{11} navíc platí $\boldsymbol{\neg Q_k(t + 1{,}m_{t+1})}$ pro každé $q_b \neq q_k \in Q$. Tím je indukční krok dokázán.

□

Na základě tohoto lemmatu již není pro binární Turingův stroj M příliš těžké dokázat korespondenci mezi validitou tvrzení f_M v teorii $\mathcal{T}_M$ a zastavení příslušného stroje na prázdném vstupu.

Značení. *Nechť* $\boldsymbol{D}$ *je nějaký model teorie* $\mathcal{T}_M$ *jazyka* $\mathcal{L}$, *potom pro každý funkční/relační symbol* f *označme* $f^{\boldsymbol{D}}$ *interpretaci* f *v* $\boldsymbol{D}$.

Tvrzení 3.2.25. *Ať* $M = (A{,}\Sigma{,}Q{,}\delta{,}q_h{,}q_s{,}\epsilon)$ *je binární Turingův stroj, kde* $A = \{\epsilon{,}s_1{,}\ldots{,}s_r\}$, $Q = \{q_s{,}q_h{,}q_1{,}q_2{,}\ldots{,}q_l\}$. *Potom* f_M *je v teorii* $\mathcal{T}_M$ *validní právě tehdy, když* M *neskončí práci na prázdném vstupu.*

Důkaz.
$(\Leftarrow)$ Nechť M neskončí práci na prázdném vstupu a uvažujme nějaký model

B teorie $\mathcal{T}_M$ s nosnou množinou B. Z předchozího lemmatu existuje pro každé $t \in \mathbb{N}$ právě jedno m_t, že platí $H(t,m_t)$. Ze stejného lemmatu také plyne, že pokud je v t-tém kroku stroj M ve stavu $q_j \in Q$, tak platí $Q_j(t,m_t)$. Ale my jsme předpokládali, že M na prázdném vstupu neskončí práci. Tedy jistě nemůže nastat $\neg f_M = Q_h(t,n) \cap H(t,n)$, $\forall t,n \in \mathbb{N}$, neboť je vždy jeden z konjugátů nepravdivý. Zbývá si tedy rozmyslet, že nemůžeme dostat $\neg f_M$ mimo minimální strukturu $\bigcup\limits_{m,t}(m,t)$.
Pro spor uvažujme $x \neq (m,t)$, $\forall m,t \in \mathbb{N}$, že platí $H(x) \cap Q_h(x)$. Z f_{10} dostaneme $s_j \in A$, že platí $S_j(x)$. Z axiomu f_{16} ovšem víme, že jsme se kromě počátku museli do každé pozice nad kterou platí relace $H(\ )$ nějak dostat na základě aplikace funkce δ. Formálně tedy nutně existují $s_{i_1} \in A$, $q_{j_1} \in Q$ a prvek $y_1 \in B$ splňující $S_{i_1}(y_1) \cap Q_{j_1}(y_1) \cap H(y_1)$ a $R_1 \in \{L, P\}$, že $\mathrm{krok}(R_1(y_1)) = x$. Z f_3,f_{10} a f_{11} jsou navíc tyto indexy určeny jednoznačně (kde R_1 je určeno jednoznačně triviálně, neboť je δ funkcí). Pokud je $R_1 = L$, tak označme $r_1 = -1$ a pokud $R_1 = P$, tak $r_1 = 1$.
Všimněme si, že nalezené y_1 splňuje $H(y_1)$. Protože je navíc y_1 vzorem prvku x při zobrazení $krok \circ R_1(\ )$, kde x je mimo minimální strukturu $\bigcup\limits_{m,t}(m,t)$, tak je mimo ní nutně také y_1. Ta samá úvaha lze tedy analogicky aplikovat na y_1. Tento postup budeme takto iterovat a dostaneme tak posloupnost čtveřic prvků $(y_k,s_{i_k},q_{j_k},R_k)$, pro nějaká $s_{i_k} \in A$, $q_{j_k} \in Q$, $y_k \in B$ a $k \in \mathbb{N}$, kde čtveřice

$$\begin{array}{ll}
(y_1,s_{i_1},q_{j_1},R_1), & \text{splňuje } S_{i_1}(y_1) \cap Q_{j_1}(y_1) \cap H(y_1) \text{ a } \delta(q_{j_1},s_{i_1}) = (q_h,s,r) \\
(y_2,s_{i_2},q_{j_2},R_2), & \text{splňuje } S_{i_2}(y_2) \cap Q_{j_2}(y_2) \cap H(y_2) \text{ a } \delta(q_{j_2},s_{i_2}) = (q_{j_1,s_{i_1}},r_{i_1}) \\
(y_3,s_{i_3},q_{j_3},R_3), & \text{splňuje } S_{i_3}(y_3) \cap Q_{j_3}(y_3) \cap H(y_3) \text{ a } \delta(q_{j_3},s_{i_3}) = (q_{j_2,s_{i_2}},r_{i_2}) \\
\vdots & \qquad\qquad\qquad\qquad \vdots
\end{array}$$

Jenomže pro Turingův stroj M nutně existuje jenom konečně mnoho dvojic $(q,s) \in Q \times A$, tzn. existují prvky $k,l \in \mathbb{N}$, $k < l$ přirozené, že $s_{i_k} = s_{i_l}$ a $q_{j_k} = q_{j_l}$. Odsud ovšem pro $k - l \geq 1$ platí $q_h = q_{i_{k-l}}$, tzn. $\delta(q_{i_{k-l}},a_{i_{k-l}})$ není definované a to je spor.

($\Rightarrow$) Tuto implikaci ukážeme obměnnou. Mějme tedy stroj M, který skončí práci na prázdném vstupu. Ukážeme, že potom existuje struktura **D**, že **D** je modelem $\mathcal{T}_M$, ale $\mathbf{D} \not\models f_M$. Definujme ji následovně. Nosnou množinu **D** zvolme $D = \mathbb{N} \times \mathbb{N}$. Obraz logických symbolů je pak definovaný následujícími body:

- $\text{počátek}^{\mathbf{D}}$=(1,1)
- $\text{krok}^{\mathbf{D}}(k,n) = (k+1,n)$
- $\text{P}^{\mathbf{D}}(k,n) = (k,n+1)$
- $\text{L}^{\mathbf{D}}(k,n) = (k, \max\{1,n-1\})$
- $\text{kraj}^{\mathbf{D}}(k,n)$, pokud $n = 1$
- $Q_i^{\mathbf{D}}(k,n)$, T je v k-tém kroku na n-té souřadnici ve stavu q_i
- $S_j^{\mathbf{D}}(k,n)$, T má v k-tém kroku na n-té souřadnici znak S_j

- vstup$^{\mathbf{D}}(k,n)$, pokud $k = 1$

Není těžké si uvědomit, že z definice Turingova stroje a z obrazu funkčních a relačních symbolů, platí nezávisle na ohodnocení axiomy $f_1 - f_{16}$. Ale my jsme předpokládali, že M na prázdném vstupu skončí práci. Tedy ve struktuře $\mathbf{D}$ neplatí f_M. Odsud $\mathbf{D} \not\models f_M$ a f_M tedy není v teorii $\mathcal{T}_M$ validní. $\square$

Abychom dotáhli dokonce zbytek důkazu, nevyhnuli bychom se zdlouhavé technické diskuzi o kódování. Pokusíme se jej zde tedy pouze naznačit. Myšlenka doufám bude z této diskuze zřejmá. Podobně jako jsme ztotožnili Turingovy stroje s jejich Turingovým číslem, tak můžeme ztotožnit formule jazyka $\mathcal{L}$ s binárními řetězci nějakým „interpretovatelným“ zobrazením ψ, splňujícím, že

$$\alpha(M) \to \psi(f_1,f_2,...,f_{16},f_M)$$

je pro každý stroj M algoritmicky řešitelné. Jinými slovy Turingovu stroji při předání popisu $\alpha(M)$ předáme všechnu informaci potřebnou k popsání teorie $\mathcal{T}_M$ a tvrzení f_M, na jehož validitu se ptáme. Kdyby nyní existoval stroj S, že

$$S(\psi(f_1,f_2,...,f_{16},f_M)) = \begin{cases} 1, & \text{pokud } \varphi \text{ je validní v } \mathcal{T}_M \\ 0, & \text{jinak} \end{cases}$$

kde φ je formule v logice $\mathcal{L}$, potom by existoval stroj U, který by na vstupu $\alpha(M)$ udělal následující:

1. Transformoval by $\alpha(M)$ do tvaru $\psi(f_1,...,f_{16},f_M)$
2. Potom by na $\psi(f_1,...,f_{16},f_M)$ pustil stroj S

Jelikož f_M je v teorii $\mathcal{T}_M$ pro binární Turingův stroj M validní právě tehdy, když M definuje výstup na prázdném vstupu, tak

$$U(\alpha(M)) = \begin{cases} 1, & \text{pokud je } M(\emptyset) \text{ definované} \\ 0, & \text{jinak} \end{cases}$$

, což je ovšem spor s neřešitelností prázdného problému zastavení.
V průběhu kapitoly jsme tedy odvodili, že existuje jazyk $\mathcal{L}$, ve kterém nelze na základě poskytnutých axiomů obecně algoritmicky rozhodnout o dokazatelnosti zvoleného tvrzení. Jinými slovy neexistuje obecný postup, který by rozhodoval o validnosti tvrzení v příslušné teorii. Formálně se tento fakt dá formulovat například následovně:

Věta 3.2.26. *(Entscheidungsproblem) Existuje jazyk $\mathcal{L}$ a interpretovatelné kódování ψ, že zobrazení U dané předpisem*

$$U(\psi(f_1,f_2...,f_r,g)) = \begin{cases} 1, & \textit{pokud } \{f_1,....,f_r\} \vdash g \\ 0, & \textit{jinak} \end{cases},$$

přes všechna $r \in \mathbb{N}_0$ a $f_1,....,f_r,g$ formule jazyka $\mathcal{L}$, není algoritmicky řešitelné.

3.3 Definice a základní poznatky

Kolmogorov roku 1965 vydal článek s návrhem alternativní míry informace. Podobně jako Shannon, Kolmogorov definoval funkci, která měří množství informace, které je nám dáno na základě nějakého pozorování, jako minimální počet bitů potřebný k popsání tohoto pozorování. V případě Kolmogorovova se však místo identifikace prvku vzhledem k pravděpodobnostnímu rozdělení jedná o délku minimální algoritmického popisu. Ta se dnes nazývá Kolmogorovovskou složitostí. Nyní už máme dostatečný matematický aparát k tomu, abychom o ní mohli mluvit formálně.

Definice 3.3.1. *Nechť T je binární Turingův stroj a $x \in \{0{,}1\}^*$, potom hodnotu:*

$$K_T(x) = \begin{cases} \min\{\ell(u) : u \in D(T), T(u) = x\}, & \text{pokud } x \in Im(T) \\ \infty, & \text{jinak} \end{cases}$$

nazveme Kolmogorovovskou složitostí x vzhledem k T.

Rádi bychom tuto definici zprostili závislosti na stroji T. Z intuice, kterou máme by nás mohlo napadnout vzít $\min\limits_T K_T(x)$ jako délku minimálního algoritmického popisu. Uvědomme si ovšem, že taková veličina by byla nulová pro každé $x \in \{0{,}1\}^*$. V příkladu 3.2.3 jsme totiž našli pro každé $x \in \{0{,}1\}^*$ stroj, který na libovolném, tedy i prázdném vstupu, definoval výstup x. Podfuk je v tom, že popis daného stroje je velice silným způsobem závislý na x. Jinými slovy je potřeba nějak vyvážit složitost algoritmu a délku vstupu. Jedním způsobem, jak tohoto docílit, je vzít Kolmogorovovskou složitost vzhledem k nějakém fixovanému univerzálnímu stroji U. Univerzální stroje jsou dostatečně složité na to, aby k takovým případům nedocházelo. Ale hlavně vezmeme-li libovolný stroj T a vstup $x \in D(T)$, potom $U(T{,}x) = T(x)$ a $\ell((T{,}x)) \approx 2\ell(P(T)) + \ell(x)$, kde P je příslušné deskripční číslo. Tedy univerzální stroj váží složitost algoritmu (ve formě délky deskripčního čísla) vůči délce vstupu toho algoritmu.

Uvědomme si navíc, že Kolmogorovská složitost vzhledem k univerzálnímu stroji je konečná na libovolném x. Univerzální stroje totiž umí simulovat každý algoritmus. Tedy speciálně jsou také schopny spustit program „Nic nedělej se vstupem a skonči práci“. Délka takového vstupu je navíc $\ell(x) + \mathcal{O}(1)$, kde $\mathcal{O}(1)$ je délka prefixu, který U potřebuje ke zmíněné simulaci. To nám dává horní odhad na Kolmogorovskou složitost vzhledem k U

Tvrzení 3.3.1. *Nechť U je univerzální Turingův stroj, potom $Im(U) = \{0{,}1\}^*$ a platí, že $K_U(x) \leq \ell(x) + \mathcal{O}(1)$ pro každé $x \in \{0{,}1\}^*$.*

Důkaz. Mějme $x \in \{0{,}1\}^*$. Ve tvrzení 3.2.2 jsme našli stroj T, který na libovolném vstupu $y \in \{0{,}1\}^*$ definoval výstup $T(y) = y$. Z poznámky 3.2.14 navíc víme, že existuje řetězec $t \in \{0{,}1\}^*$ splňující $U(t \circ y) = T(y)$ pro každé $y \in D(T)$. Odsud $t \circ x \in \{y \in D(U) : U(y) = x\}$. Tedy $x \in Im(U)$. Z definice Kolmogorovovské složitosti je navíc

$$K_U(x) = \min \underbrace{\{\ell(y) : y \in D(U), U(y) = x\}}_{\neq \emptyset}$$

Ale minimum neprázdné množiny prvků $\mathbb{N}_0$ je také prvkem $\mathbb{N}_0$ a to jsme chtěli. $\square$

Zdaleka však nejdůležitějším argumentem pro volbu univerzálního Turingova stroje je následující vlastnost invariantnosti Kolmogorovské složitosti. Jelikož se univerzální Turingovy stroje umějí simulovat navzájem, tak je nakonec v podstatě jedno, který z nich vybereme. Jejich hodnota se na každém řetězci bude lišit nejvýše o konstantu nezávisející na vstupu. Li a Vitányi v knize (Li a Vitányi, 2008), která je široce přijímána jako základní literatura ke Kolmogorovské složitosti, zmiňují, že tato vlastnost je hlavním důvodem, proč si Kolmogorovovská složitost vydobyla své postavení.

Věta 3.3.2. *(Invariantnost kolmogorovovské složitosti) Nechť U, T jsou Turingovy stroje a ať U je navíc univerzální. Potom existuje konstanta $c \in \mathbb{Z}$, že*

$$K_U(x) \leq K_T(x) + c, \forall x \in D(T)$$

Důkaz. U je univerzální, tedy existuje $t \in \{0,1\}^*$, že $U(t \circ z) = T(z), \forall z \in D(T)$. Mějme nyní $x \in Im(T)$ pevné. Protože $x \in Im(T)$, tak je hodnota $K_T(x) \in \mathbb{N}_0$. Ať tedy $y \in D(T)$ splňuje vztahy $T(y) = x$ a $K_T(x) = \ell(y)$. Potom z univerzality U a volby t plyne $U(t \circ y) = x$. To ovšem znamená, že

$$t \circ y \in \{z : z \in D(U), U(z) = x\}$$

a odsud konečně:

$$K_U(x) = \min\{\ell(z) : z \in D(U), U(z) = x\} \leq \ell(t \circ y) = \ell(t) + \ell(y) = \ell(t) + K_T(x)$$

□

Důsledek 3.3.3. *Ať U_1, U_2 jsou univerzální Turingovy stroje. Potom existuje konstanta $c \in \mathbb{N}_0$, že*

$$|K_{U_1}(x) - K_{U_2}(x)| \leq c, \forall x \in \{0,1\}^*$$

Důkaz. Jedná se o okamžitý důsledek věty 3.3.2. □

Definice 3.3.2. *Zafixujme nějaký univerzální stroj U. Potom pro každé $x \in \{0,1\}^*$ nazveme číslo*

$$K(x) := K_U(x)$$

Kolmogorovovskou složitostí x. O stroji U navíc budeme mluvit jako o fixním stroji.

Úmluva. *Fixní stroj z definice 3.3.2 budeme nadále značit $\mathcal{U}$*

Podobně jako Shannonova teorie se Kolmogorovova teorie točí okolo komprese. Zkoumání komprese z hlediska algoritmické složitosti má tu výhodu, že ve většině případů nás komprese stejně zajímá kvůli aplikaci na počítačovém softwaru. Tedy stačí nám se omezit na řešitelná zobrazení. Uvědomme si, navíc, že Kolmogorovovská složitost stojí na faktu, že jakákoliv pravidelnost nebo struktura v řetězci se dá využít k tomu, abychom jej nějak popsali. To speciálně platí i pro nějakou strukturu způsobenou zdrojem na základě pravděpodobnostního rozdělení (o tom více v následující kapitole).
Tento „nestatistický“ izolovaný pohled na kompresi dal za vznik celé řadě nových algoritmů. Jako hlavní příklad by mohl sloužit tzv. Lempel Zivův algoritmus, z nějž mimo jiné vycházejí jedny z dnes nejpoužívanější komprimačních formátů *.gzip*, *.zip*.

Definice 3.3.3. *Buď $c \in \mathbb{N}_0$ a $x \in \{0,1\}^*$. Potom řekneme, že x je c-nekomprimovatelný, pokud*

$$K(x) \geq \ell(x) - c$$

x je navíc nekomprimovatelný, pokud je 0-nekomprimovatelný.

Je dobré si uvědomit, že existují řetězce, které jdou zkomprimovat velice příjemně, například se jedná o řetězce tvaru $1^{(n)}$. Komprimační algoritmus nepotřebuje abychom mu předali nic jiného než délku sekvence, což jsme schopni udělat v přibližně $\log_2(n)$ bitech (formálně ukážeme v lemmatu 4.1.6). Tedy $K(1^{(n)}) \leq \log_2(n) + \mathcal{O}(1)$. Ovšem i přes existenci takovýchto řetězců, je absolutní většina binárních posloupností v podstatě nekomprimovatelná. K důkazu tohoto tvrzení nám stačí poměrně jednoduchý kombinatorický argument.

Tvrzení 3.3.4. *Nechť $n \in \mathbb{N}_0$. Potom počet řetězců délky n, které jsou c nekomprimovatelné pro nějakou konstantu $c \in \mathbb{N}_0$ je vyšší než*

$$2^n - 2^{n-c}$$

Jinými slovy poměr c-nekomprimovatelných řetězců vůči všem řetězcům délky n je vyšší než $1 - 2^{-c}$.
Speciálně existuje alespoň jeden nekomprimovatelný řetězec pro každé $n \in \mathbb{N}$

Důkaz. Počet řetězců délky n které jdou zkomprimovat pod $n - c$ bitů nemůže být větší než počet vstupů nejvyšší možné délky $n - c - 1$

$$\sum_{i=0}^{n-c-1} 2^i = 2^{n-c} - 1$$

Všechny zbylé řetězce délky n jsou c-nekomprimovatelné, tedy jich je alespoň $2^n - (2^{n-c}) + 1$. Speciálně pro $c = 0$ tedy máme alespoň jeden nekomprimovatelný řetězec. Pokud jejich počet podělíme počtem všech možných řetězců délky n, tak nám vyjde poměr $1 - 2^{-c} + \frac{1}{2^n} > 1 - 2^{-c}$. ☐

Jak tuto větu interpretovat? Nejprve si lze všimnout, že existuje alespoň jedna nekomprimovatelná posloupnost. To znamená posloupnost bez nějaké využitelné vnitřní struktury. Dále je pro každou délku nejméně polovina ze zpráv zkomprimovat nejvýše o jeden bit. Více jak 3/4 z nich nejvýše o dva bity, více jak 1023/1024 méně než deset bitů atd.
Kolmogorovovská složitost se z dosavadního textu může zdát jako poměrně užitečný nástroj pro využití v praktických aplikacích. Má to však svůj háček. Kolmogorovovská složitost totiž nelze algoritmicky spočítat. Samozřejmě vždycky můžeme najít nějaký test, který se kouká po využitelných pravidelnostech, ale nikdy si nemůžeme být jisti, že vzor který najdeme je opravdu ten nejvýhodnější. Tedy Kolmogorovovskou složitost lze obecně jenom odhadnout shora. Někoho by samozřejmě mohlo napadnout, pouštět stroj $\mathcal{U}$ postupně na všechny řetězce v lexikografickém pořadí a zkoumat, jestli definují kýžený výstup. Nikde však není zaručené, že stroj na daných řetězcích bude nějaký výstup vůbec definovat. Jak už víme z problému zastavení, tak na zjištění, zda-li stroj na určitém vstupu někdy skončí práci, jsme krátcí.
Důkaz tohoto tvrzení provedeme sporem. Všimněme si nápadné podoby tohoto důkazu s Berryho pradoxem.

Věta 3.3.5. *Zobrazení $b \to (K(b))_2$ není algoritmicky řešitelné. Tedy neexistuje stroj T takový, že pro každé $b \in \{0,1\}^*$*

$$T(b) = (K(b))_2$$

Důkaz. Tvrzení ukážeme sporem. Nechť existuje stroj M, který na libovolném vstupu $x \in \{0,1\}^*$ dá výstup $(K(x))_2$. Za chvíli zkonstruujeme stroj S, který bude M používat jako knihovní funkci a který na libovolném vstupu $(n)_2$, $n \in \mathbb{N}$ dá výstup $m^{(n)}$, kde $m^{(n)}$ je lexikograficky první prvek $\{0,1\}^*$ splňující $K(m^{(n)}) > n$. Z univerzality fixního stroje $\mathcal{U}$ pak plyne:

$$n < K(x) \leq \ell(n) + c = \log_2(n) + \mathcal{O}(1)$$

Z asymptotiky n a $\log_2(n) + \mathcal{O}(1)$ by ovšem takto pro n dostatečně velké platilo $n \geq \log_2(n) + \mathcal{O}(1) = \ell(n)$. Což by byl spor. Zbývá tedy ukázat, že existence M implikuje existenci S. Práci S opět popíšeme algoritmicky:
S na vstupu dostane $(n)_2$. Pásku si na začátku 3. kroku vždy bude udržovat ve tvaru: $* \ (n)_2 \ |_1 \ \gamma \ |_2 \ (K(\gamma))_2 \ |_3$, kde na spočítání $(K(\gamma))_2$ volá v kroku 2. stroj S. V krocích 3. a 4. pak porovnává, jestli je $(K(\gamma))_2$ lexikograficky větší než $(n)_2$. Třetí krok porovnává jejich délku a jestliže jsou stejně dlouhé, tak krok 4. zjišťuje, který z nich zapisuje větší číslo. V případech, kdy $(K(\gamma))_2 < (n)_2$, S načítá lexikograficky následující řetězec v kroku 5.. Podrobně:

1. S transformuje pásku do tvaru $* \ n_2 \ |_1 \ \epsilon \ |_2 \ \epsilon \ |_3$
2. S pustí stroj M na úseku izolovaném $|_2$ a $|_3$ a výsledek upraví tak, aby mezi 0,1 nebyly žádné jiné znaky.
3. Na pásce je nyní $* \ \alpha \ |_1 \ \beta \ |_2 \ \gamma \ |_3$. Nechť a je první neoznačený prvek za $*$ a b je první neoznačený prvek za $|_2$. Jestliže
   (a) $a = |_1$ a
      i. $b = |_3$, tak S vše odznačí a pokračuje krokem 4.
      ii. $b \neq |_3$, tak S smaže vše za $|_2$ a vše před $|_1$. Potom skončí práci.
   (b) $a \neq |_1$ a
      i. $b = |_3$, tak S vše odznačí a pokračuje krokem 5.
      ii. $b \neq |_3$, tak S označí a i b a pokračuje krokem 3.
4. Nechť a je první neoznačený prvek za $*$ a b je neoznačený prvek za $|_2$. Jestliže
   (a) $a = 0$ a
      i. $b = 0$, tak S označí a i b a pokračuje krokem 4.
      ii. $b = 1$, tak S smaže vše za $|_2$ a vše před $|_1$. Potom skončí práci.
   (b) $a = 1$ a
      i. $b = 0$, tak S vše odznačí a pokračuje krokem 5.
      ii. $b = 1$, tak S označí a i b a pokračuje krokem 4.
5. Nechť a je první označený prvek od $|_2$ doleva. Jestliže

(a) $a = 1$, tak jej S nahradí $\dot{0}$ a pokračuje krokem 5.

(b) $a = 0$, tak jej S nahradí 1, vše na pásce odznačí a pokračuje krokem 3.

(c) $a = |_1$, tak S posune vstup o jeden prvek doprava a na uvolněné místo vloží 0. Vše mezi $|_1$ a ${}_2$ změní na 0. Pokračuje krokem 3.

□

Na tomto místě je dobré zmínit, že v literatuře se pracuje i s formami Kolmogorovovské složitosti, které řešitelné jsou. Pokud kupříkladu při definici Kolmogorovovské složitosti nějak penalizujete každý krok, který univerzální stroj při výpočtu udělá, tak vám může vzniknout něco, co už řešitelné je. Z jiného pohledu je možné zase Kolmogorovovskou složitost omezit jen na nějaký určitý výsek algoritmů. Jedno takové odvětví výzkumu s bohatou historií se v literatuře vyskytuje pod pojmem Lempel-Zivova složitost.

Jeden z amerických profesorů fyziky působících například na Harvardské univerzitě, Charles H. Benett, namítnul (Bennett, 2003), že více než se složitostí má Kolmogorovovská složitost co dočinění s náhodou. Když se díváme na nějaký řetězec a řekneme, že vypadá náhodně, tak tím obvykle myslíme, že ho nejsme schopni nějak snadno popsat. Jinými slovy jej neumíme zařadit do nějakého úzkého výseku všech možných prvků, například $\{a^{(n)}, a \in \{0{,}1\}\}, n \in \mathbb{N}$. V zásadě tak tedy mluvíme o nekomprimovatelných řetězcích. Ty mají relativně k délce velkou Kolmogorovovoskou složitost. Náhoda však nekoresponduje s naší intuicí o složitosti. Opravdu složitý řetězec, by měl být někde na půli cesty mezi triviálním a zcela náhodným. Když vyjdeme z této intuice, tak předcházející tvrzení říká něco poměrně zajímavého o náhodě. O některých řetězcích jsme schopni ihned říct, že náhodné nejsou. U jiných řetězců žádný vzor nenajdeme. Přesto nelze s jistotou určit, že neexistuje nějaké pravidlo, které nám zůstává skryté. Například zapíšeme-li binárně nějaký výsek rozvoje čísla π, tak výsledný řetězec projde většinou dnešních testů náhodnosti.

$$\underbrace{11}_{3}\underbrace{1}_{1}\underbrace{100}_{4}\underbrace{1}_{1}\underbrace{101}_{5}\underbrace{1001}_{9}$$

Asi nikdo by ovšem rozvoj π neoznačil za náhodný. Pomocí Taylorova polynomu jsme totiž schopni napsat pravidlo, které předpoví libovolnou jeho číslici. Celá myšlenka náhody však souvisí s nepředvídatelností.

Pro zajímavost zmiňme, že Benett definoval alternativní míru složitosti, která je blíž naší intuici. Místo toho, aby v ní pracoval s délkou nejkratšího řetězce, tak Benett navrhnul, aby se složitost definovala jako počet kroků, které univerzální stroj potřebuje při výpočtu na onom nejkratším vstupu. Takto pracujeme se složitostí pravidla spíše než s délkou popisu. Jelikož navíc pro zcela náhodné řetězce vpodstatě neexistuje lepší popis než prosté vyjmenování číslic, tak při tom univerzální stroj neudělá mnoho kroků. To řeší náš problém. Benettova definice se dá dohledat pod pojmem Benettova logická hloubka.

4. Vztah

4.1 Zvonkinova věta

Shannon i Kolmogorov se snažili postihnout kvantitu informace uložené ve zprávě na základě jakéhosi vzoru s ní spojenou. Shannon pracoval se zprávu jako s prvkem nějakého rozdělní. Jeho definice stála na pravidelnosti toho, jaké zprávy produkuje zdroj, ze kterého daná zpráva pochází. Informaci tedy vnímal v jakési globální rovině. Kolmogorova definice na druhou stranu kvantifikovala informaci dané zprávy izolovaně. Informace u něj v jádru měřila sílu nějakého skrytého vzoru, který tuto zprávu popisoval. Je krásným důsledkem, že přijmeme-li nějaké základní předpoklady Shannonovy teorie, tak tyto definice pocházející ze dvou protipólních pohledů na daný prvek, vycházejí až na zanedbatelnou chybu asymptoticky stejně. Formálně to znamená, že $K(M_n) \approx I(M_n)$. Tento vztah se pokusíme dokázat v této kapitole. V podstatě jej budeme redukovat na dvě nerovnosti, na $P(\frac{K(M_n)}{n} \geq H(X) - \epsilon) \xrightarrow{n\to\infty} 1$ a $P(\frac{K(M_n)}{n} \leq H(X) + \epsilon) \xrightarrow{n\to\infty} 1$. Spodní odhad získáme poměrně snadno z tvrzení 2.3.5, kde jsem dokázali, že pro každé kódování ψ platí $nH(X) \preceq \ell(\psi(M_n))$. O něco technicky náročnější je důkaz horního odhadu. Myšlenka je však poměrně jednoduchá. V průběhu kapitoly zkonstruujeme stroj, který na nějakém vstupu $\rho(m)$ bude definovat výstup m, pro $m \in Im(M_n), n \in \mathbb{N}$ libovolné. Tento vstup bude mít ze Stirlingovy věty příznivé asymptotické chování. To ukážeme ve větě 4.1.4. Ve chvíli, kdy necháme univerzální stroj simulovat interpretaci tohoto vstupu, tak získáme horní odhad.

Definice 4.1.1. *Nechť M_n je náhodná zpráva emitována zdrojovou veličinou X, kde $Im(X) = \{a_1,a_2,...,a_v\}$. Řekneme, že zprávy $m_1,m_2 \in Im(M_n)$ mají stejný typ, pokud počet a_i ve zprávě m_1 je stejný jako počet a_i ve zprávě m_2, pro každé $i = 1,2,...,v$.*

Úmluva. *Po zbytek kapitoly zafixujme veličinu $X, Im(X) = \{a_1,a_2,...,a_v\}$, kde $v \geq 2$*

Značení. *Buď M_n náhodná zpráva emitována zdrojovou veličinou X. Mějme navíc $m \in Im(M_n)$. Potom označíme,*

$$\mathsf{\rho}(m) :=< (n)_2,(p)_2,(s_1)_2,(s_2)_2,...,(s_v)_2 >,$$

kde s_i je počet a_i ve zprávě m pro $i = 1,2,...,v$ a p je lexikografické pořadí m mezi zprávami stejného typu. (<,> zde označuje párovací funkci z definice 2.3.4)

V předchozím textu jsme zmínili Stirlingovu větu. Jelikož se jedná o tvrzení, jehož důkaz je spíše technického charakteru a které je navíc všeobecně známé, tak zde jeho důkaz ukazovat nebudeme.

Lemma 4.1.1. *(Slabá Stirlingova věta) Pro každé $n \in \mathbb{N}$ platí $(\frac{n}{e})^n \leq n! \leq n(\frac{n}{e})^n$*

(Cover a Thomas, 2006, Problem 11.20)

Představme si, že máme nějakých n prvků, přičemž víme, že každý z nich je právě jednoho z v typů. Označme s_i počet prvků $i-$tého typu v této množině. Víme, že počet různých permutací této množiny, bereme-li prvky v rámci jedné třídy jako nerozlišitelné, je právě $\frac{n!}{s_1!s_2!...s_v!}$. Stirlingova formule nám dává nástroj, jak shora odhadnout počet různých permutací této množiny a tím mimo jiné odhadnout kolik bitů je třeba na její binární zápis.

Tvrzení 4.1.2. *Nechť* $n, s_1, s_2, ..., s_v \in \mathbb{N}$*, že* $s_1 + s_2 + ... + s_v = n$*. Potom*

$$\log_2\left(\frac{n!}{(s_1)!(s_2)!...(s_v)!}\right) \leq o(n) - n\sum_{i=1}^{v} \frac{s_i}{n} \log_2(\frac{s_i}{n})$$

Důkaz. Položme $r_i := \frac{s_i}{n}$, pak z lemmatu 4.1.1 platí

$$\begin{aligned}
\frac{n!}{(s_1)!(s_2)!...(s_v)!} &= \frac{n!}{(nr_1)!(nr_2)!...(nr_v)!} \\
&\leq \frac{(\frac{n}{e})^n n}{(\frac{nr_1}{e})^{nr_1}(\frac{nr_2}{e})^{nr_2}...(\frac{nr_v}{e})^{nr_v}} \\
&= \frac{(\frac{n}{e})^n}{(\frac{n}{e})^{n(r_1+r_2+...+r_v)}} \cdot n \cdot \frac{1}{\prod\limits_{i=1}^{v} r_i^{nr_i}} \\
&= 1 \cdot n \cdot 2^{-\log_2(\prod\limits_{i=1}^{v} r_i^{nr_i})} \\
&= n \cdot 2^{-n\sum\limits_{i=1}^{v} r_i \log_2(r_i)} \\
&= n \cdot 2^{-n\sum\limits_{i=1}^{v} \frac{s_i}{n} \log_2(\frac{s_i}{n})}
\end{aligned}$$

Protože obě krajní funkce tohoto řetězce nerovností jsou kladné, pak tuto nerovnost můžeme zlogaritmovat. Nebo-li platí:
$\log_2\left(\frac{n!}{(s_1)!(s_2)!...(s_v)!}\right) \leq \log_2(n) - n\sum\limits_{i=1}^{v} \frac{s_i}{n}\log_2(\frac{s_i}{n}) \leq o(n) - n\sum\limits_{i=1}^{v} \frac{s_i}{n}\log_2(\frac{s_i}{n})$
a to jsme chtěli. ☐

Toto tvrzení použijeme k tomu, abychom shora odhadli asymptotické chování vstupu $\rho(m), m \in Im(M_n)$ v pravděpodobnosti. Konkrétně ukážeme, že

$$P\left(\frac{\ell(\rho(M_n))}{nH(X)} \leq 1 + \epsilon\right) \to 1$$

Připomeňme, že délka $\rho(m)$ závisí v podstatě jenom na třech věcech: na délce n, počtech prvků $i-$tých písmene s_i a na pořadí m mezi zprávami stejného typu, p. Není těžké si všimnout, že $s_i \leq n, i = 1,2,...,v$, přičemž rovnost nastává právě tehdy když m je složeno pouze z jednoho písmene. Dále si uvědomme, že na m se dá koukat jako na posloupnost prvků roztříditelných do v různých tříd, v rámci nichž, jsou nerozlišitelné. Pořadí zprávy m mezi zprávami stejného typu tedy nemůže přesáhnout počet různých permutací této posloupnosti, ten je ovšem roven $\frac{n!}{s_1!...s_v!}$. Na to kolik nul a jedniček je třeba na zapsání tohoto čísla jsme získali odhad právě v minulém tvrzení. To nám dává poměrně elegantní výsledek:

$$\rho(m) \leq (v+1)(\log_2(n) + o(\log_2(n))) - \frac{s_i}{n}\log_2(\frac{s_i}{n}) + o\left(\frac{s_i}{n}\log_2(\frac{s_i}{n})\right)$$

Mějme náhodné veličiny $S_i^{(n)}$ značící počet prvků $i-$tého písmene v náhodné zprávě M_n a P^n pořadí M_n mezi zprávami stejného typu. Na základě předchozí diskuze se pokusíme dokázat, že pro každé $\epsilon > 0$:

$$P(\frac{\ell(\rho(M_n))}{nH(X)} \leq 1 + \epsilon) \xrightarrow{n\to\infty} 1.$$

Před tím je však vhodné uvědomit si následující jednoduché lemma.

Tvrzení 4.1.3. *Nechť M_n je náhodná zpráva se zdrojovou veličinou X. Mějme náhodné veličiny $S_i^{(n)} := \#a_i$ ve zprávě M_n. Potom platí že*

$$-\sum_{i=1}^{v} \frac{S_i^{(n)}}{n} \log_2(\frac{S_i^{(n)}}{n}) \xrightarrow{P} H(X)$$

Důkaz. Z lemmatu 2.2.5 víme, že se empirické četnosti u náhodného výběru blíží jejich teoretickým pravděpodobnostem, neboli platí $\frac{S_i^{(n)}}{n} \xrightarrow{P} P(X_i = a_i)$. Z věty o spojité transformaci (tvrzení 2.2.3) pro $-y\log_2(y)$ navíc dostaneme, že

$$\frac{S_i^{(n)}}{n} \log_2(\frac{S_i^{(n)}}{n}) \xrightarrow{P} P(X = a_i)\log_2(P(X = a_i))$$

a opakovaným použitím Sluckého věty pak platí:

$$-\sum_{i=1}^{v} \frac{S_i^{(n)}}{n} \log_2(\frac{S_i^{(n)}}{n}) \xrightarrow{P} -\sum_{i=1}^{v} P(X = a_i)\log_2(P(X = a_i)) = H(X)$$

□

Tvrzení 4.1.4. *Nechť M_n je náhodná zpráva se zdrojovou veličinou X. Mějme náhodné veličiny $S_i^{(n)} := \#a_i$ ve zprávě M_n, $i = 1,2,..,v$ a $P :=$ lexikografické pořadí M_n vůči zprávám stejného typu. Potom pro každé $\epsilon > 0$*

$$P\Big(\frac{\ell(\rho(M_n))}{nH(X)} \leq 1 + \epsilon\Big) \xrightarrow{n\to\infty} 1$$

Důkaz. Nejprve si uvědomme, že pro každé $n \in \mathbb{N}$ platí:

$$\begin{aligned}
\ell(\rho(M_n)) &= o(\log_2(n)) + \log_2(n) + o(\log_2(P)) + \log_2(P) \\
&\quad + \sum_{i=1}^{v} \Big(o(\log_2(S_i^{(n)})) + \log_2(S_i^{(n)})\Big) \\
&\leq o(\log_2(n)) + (v+1)\log_2(n) + o(\log_2(P)) + \log_2(P) \\
&\leq o(n) + o\Big(\log_2\Big(\frac{n!}{(S_1^{(n)})!(S_2^{(n)})!...(S_v^{(n)})!}\Big)\Big) + \log_2\Big(\frac{n!}{(S_1^{(n)})!(S_2^{(n)})!...(S_v^{(n)})!}\Big) \\
&\overset{4.1.2}{\leq} o(n) + o\Big(-n\sum_{i=1}^{v} \frac{S_i^{(n)}}{n} \log_2\Big(\frac{S_i^{(n)}}{n}\Big)\Big) - n\sum_{i=1}^{v} \frac{S_i^{(n)}}{n} \log_2\Big(\frac{S_i^{(n)}}{n}\Big)
\end{aligned}$$

Tedy z lemmatu 4.1.3 a Sluckého věty platí:

$$\begin{aligned}
\frac{\ell(\rho(M_n))}{nH(X)} &\leq \frac{o(n)}{nH(X)} + \frac{o\Big(-n\sum\limits_{i=1}^{v}\frac{S_i^{(n)}}{n}\log_2(\frac{S_i^{(n)}}{n})\Big)}{nH(X)} + \frac{-n\sum\limits_{i=1}^{v}\frac{S_i^{(n)}}{n}\log_2(\frac{S_i^{(n)}}{n})}{nH(X)} \\
&= \frac{o(n)}{nH(X)} + \frac{o\Big(-n\sum\limits_{i=1}^{v}\frac{S_i^{(n)}}{n}\log_2(\frac{S_i^{(n)}}{n})\Big)}{-n\sum\limits_{i=1}^{v}\frac{S_i^{(n)}}{n}\log_2(\frac{S_i^{(n)}}{n})} \cdot \frac{-n\sum\limits_{i=1}^{v}\frac{S_i^{(n)}}{n}\log_2(\frac{S_i^{(n)}}{n})}{nH(X)} \\
&\quad + \frac{-n\sum\limits_{i=1}^{v}\frac{S_i^{(n)}}{n}\log_2(\frac{S_i^{(n)}}{n})}{nH(X)} \\
&\xrightarrow{P} 0 + 0 \cdot 1 + 1
\end{aligned}$$

A konečně odsud

$$\begin{aligned}
1 &\geq P\Big(\frac{\ell(\rho(M_n))}{nH(X)} \leq 1+\epsilon\Big) \\
&\geq P\Big(\frac{o(n)}{nH(X)} + \frac{o\Big(-n\sum\limits_{i=1}^{v}\frac{S_i^{(n)}}{n}\log_2(\frac{S_i^{(n)}}{n})\Big)}{nH(X)} + \frac{-n\sum\limits_{i=1}^{v}\frac{S_i^{(n)}}{n}\log_2(\frac{S_i^{(n)}}{n})}{nH(X)} \leq 1+\epsilon\Big) \\
&\geq P\Big(\Big|\frac{o(n)}{nH(X)} + \frac{o\Big(-n\sum\limits_{i=1}^{v}\frac{S_i^{(n)}}{n}\log_2(\frac{S_i^{(n)}}{n})\Big)}{nH(X)} + \frac{-n\sum\limits_{i=1}^{v}\frac{S_i^{(n)}}{n}\log_2(\frac{S_i^{(n)}}{n})}{nH(X)} - 1\Big| \leq \epsilon\Big) \\
&\xrightarrow{n\to\infty} 1
\end{aligned}$$

□

Důsledek 4.1.5. *Platí, že*

$$\ell(\rho(M_n)) \approx nH(X)$$

Speciálně je ρ *ideálním kódováním.*

Důkaz. Z tvrzení 4.1.4 víme, že $\ell(\rho(M_n)) \preceq nH(X)$. Ale z věty 2.3.6 $nH(X) \preceq \ell(\rho(M_n))$. Z tvrzení 2.2.7 tedy $\ell(\rho(M_n)) \approx nH(X)$. Kódování ρ je z tvrzení 2.3.11 tedy ideální. □

Ještě je třeba ukázat, že zobrazení $\rho(m) \to m$ je řešitelné. Algoritmus, který popíšeme bude používat dvě dílčí konstrukce:

1. V lemmatu 4.1.6 ukážeme, že Turingovy stroje jsou schopné interpretovat, binární zápis přirozených čísel, neboli že $(n)_2 \to 1^{(n)}$ je řešitelné.
2. V lemmatu 4.1.7 pak zkonstruujeme stroj řešící zobrazení

$$<\alpha_1,\alpha_2,\ldots,\alpha_m> \to *\alpha_1,\alpha_2,\ldots,\alpha_m$$

Lemma 4.1.6. *(Existence interpretačního stroje) Existuje stroj* T*, který na libovolném vstupu* $(i)_2, i \in \mathbb{N}_0$ *dává výstupní konfiguraci s řetězcem*

$$\underbrace{11\ldots1}_{i}\,\bar{\epsilon}$$

pro nějaké $\bar{\epsilon} \in \{\epsilon\}^*$

Důkaz. Stroj T bude používat pracovní abecedu $\{0,1,\dot{0},\dot{1},X,Y,Z\}$. Páska stroje T bude tvaru $X\alpha Y\beta Z11...1$. Mezi X a Y bude T udržovat vstupní řetězec α. Mezi Y a Z bude mít nejpve 0 a za Z pouze ϵ. V kroku 2. vžy porovná rovnost řetězců mezi X a Y a Y a Z. Jestliže budou rozdílné, pak T v kroku 3. přičte k binárnímu číslu zapsanému mezi Y a Z jedničku a napíše si jednu 1 za Z. Takto bude pokračovat dokud se tyto dva řetězce nebudou rovnat. Jinými slovy T vyzkouší binární zápisy všech čísel od 0 až po α a za každé z nich si napíše za Z jednu 1.

1. T vloží na konec vstupního řetězce řetězec '$Y0Z$', pak celý konfigurační řetězec posune doprava a na uvolněné místo vloží X.
2. Na pásce je nyní řetězec '$X\alpha Y\beta Z\underbrace{11...1}_{\in\mathbb{N}_0}$', kde $\alpha \in \{0,1\}^*$, $\beta\{0,1,\dot{0},\dot{1}\}^*$. Nechť a je první neoznačený prvek za X a b je první neoznačený prvek za Y. Pokud
   (a) $a = b$, pak T označí a i b a pokračuje krokem 2.
   (b) $a \neq b$ tak pokud
      i. $a = Y, b = Z$, potom T nahradí vše od Z (včetně) doleva ϵ, řetězec 1 posune na začátek pásky a skončí práci.
      ii. $a \neq Y$ nebo $b \neq Z$, potom T odznačí všechny označené prvky na pásce a pokračuje krokem 3.
3. Nechť a je první neoznačený prvek od Z doleva. Pokud
   (a) $a = 1$, pak místo něj T vloží $\dot{0}$. Pokračuje krokem 3.
   (b) $a = 0$, pak místo něj T vloží 1, odznačí všechny prvky na pásce, místo prvního ϵ na pásce vloží 1 a pokračuje krokem 2.
   (c) $a = Y$, pak T posune celý řetězec začínající na pravém sousedovi Y o jeden prvek doprava a na uvolněné místo vloží 1. Všechny prvky mezi Y a Z odznačí, Místo prvního ϵ na pásce vloží 1 a pokračuje krokem 2.

□

Příklad 4.1.1. *Ilustrujme práci stroje z lemmatu 4.1.6 na vstupu* 101
101
$X101Y0Z$
$X101Y1Z1$
$X101Y10Z11$
$X101Y11Z111$
$X101Y100Z1111$
$X101Y101Z11111$
$11111\epsilon\epsilon\epsilon\epsilon\epsilon\epsilon\epsilon\epsilon\epsilon$

Lemma 4.1.7. *Existuje stroj T, který na libovolném vstupu $< \alpha_1, \alpha_2,...,\alpha_m >$, kde $\alpha_i \in \{0,1\}^*$, definuje výstupní konfiguraci s řetězcem*

$$* \ \alpha_1 \ , \ \alpha_2 \ , \ ... \ , \ \alpha_m \ \bar{\epsilon}$$

pro nějaké $\bar{\epsilon} \in \{\epsilon\}^*$

Důkaz. Binární stroj T bude používat pracovní abecedu $\{*,\dot{1},\dot{0},0,1\}$. Při práci bude postupně analyzovat každý vstup párovací funkce. V kroku 2. zjistí, kde končí binární zápis délky interpretovaného podřetězce. Ve 3. kroku tuto délku expanduje na příslušný počet 1, aby ve 4. kroku zjistil, kde daný vstup párovací funkce končí. Potom jej v kroku 5. oddělí od zbytku vstupu čárkou. Takto pokračuje dokud nenarazí na konec vstupu.

1. Dvakrát posune vstup a na uvolněné místo vloží řetězec '$*$,'
2. Na pásce je nyní 'α , β', kde $\alpha \in \{*,0,1\}^*$,$\beta \in \{0,1,\dot{0},\dot{1}\}^*$. Nechť a je první neoznačený prvek β (zleva). Jestliže
   (a) $a \neq 0$, potom T označí a a první neoznačený prvek za nejbližší 0 od a doprava. Pokračuje krokem 2..
   (b) $a = 0$, pak T posune řetězec začínající na pravém sousedovi a vedle nejbližší ',' od a doleva. Pokračuje krokem 3..
3. Na pásce je nyní řetězec 'α , β γ', kde $\alpha \in \{*,0,1,\text{','}\}^*$, $\beta \in \{\dot{0},\dot{1}\}^*$ a $\gamma \in \{0,1\}^*$. T posune γ a na uvolněné místo vloží $|$. Potom odznačí γ. Expanduje řetězec mezi , a $|$ (použije lemma a na konci toho řetězce jsou nějaké ϵ jichž se zbaví posunutím doleva). Pokračuje krokem 4.
4. Na páce je nyní 'α , β $|$ γ', kde $\alpha \in \{*,0,1\}^*$,$\beta \in \{1\}^*$,$\gamma \in \{0,1\}^*$. Nechť a je první neznačený prvek řetězce 'β $|$ '. Jestliže
   (a) $a \neq |$, pak T označí a a první neoznačený prvek za $|$.
   (b) $a = |$, pak T posune řetězec začínající na pravém sousedovi a vedle nejbližší , od a doleva.
5. Na pásce je nyní 'α , β γ', kde $\alpha \in \{\text{','}\ ,0,1\}^*$,$\gamma \in \{0,1\}^*$, $\beta \in \{\dot{0},\dot{1}\}^*$.
   (a) $\gamma \neq \emptyset$, potom posune γ doprava, na uvolněné místo vloží ',' a odznačí všechny prvky γ.
   (b) $\gamma = \emptyset$, Tak páska je tvaru '$*$,δ', kde $\delta \in \{0,1,\text{','}\}^*$. Tedy posune δ vedle $*$ a všechny prvky na pásce odznačí.

□

Příklad 4.1.2. *Ilustujme práci stroje z lemmatu 4.1.7 na vstupu*
$< 11001,1 >$:
1110101110011011
$*,1110101110011011$
$*,\dot{1}\dot{1}\dot{1}0\dot{1}\dot{0}\dot{1}110011011$
$*,\dot{1}\dot{0}\dot{1}110011011$
$*,101|110011011$
$*,11111|110011011$
$*,\dot{1}\dot{1}\dot{1}\dot{1}\dot{1}|\dot{1}\dot{1}\dot{0}\dot{0}\dot{1}1011$
$*,\dot{1}\dot{1}\dot{0}\dot{0}\dot{1}1011$
$*,11001,1011$
$*,11001,\dot{1}0\dot{1}1$
$*,11001,1|1$

$\ast,11001,1|1$
$\ast,11001,1$
$\ast 11001,1$

Lemma 4.1.8. *(Existence interpretačního stroje) Nechť* $\{\beta_0,\beta_1,...,\beta_v\} = B$, *kde* $\beta_i \in \{0,1\}^*$ *a* $m \in B^*$, *potom existuje stroj, který na vstupu*

$$\rho(m) =< (n)_2,(p)_2,(s_1)_2,(s_2)_2,...,(s_v)_2 >$$

dá výstup $T(\rho(m)) = m$

Důkaz. T bude používat pracovní abecedu $A = \{|_{s_1},|_{s_2},...,|_{s_v},0,1,\dot{1},\dot{0},1,0,a_1,a_2,...,a_v\}$. Při výpočtu potom bude postupně procházet všechny řetězce délky n, počínaje $a_1^{(n)}$. V prvních třech krocích si stroj T upraví pásku do tvaru

$$\ast\ 1^{(n)}\triangle\ 1^{(p)}\ |_{s_1}1^{(s_1)}\ |_{s_2}1^{(s_2)}\ |_{s_3}...\ |_{s_v}1^{(s_v)}\ @$$

Kde za @ bude generovat všechny možné posloupnosti prvků$\{a_1,...,a_v\}$ délky n v lexikografickém pořadí (uvažujem $a_i < a_j$ pro $i < j$). V kroce 4.(a) vždy zkontroluje, jestli má aktuální řetězec s_i exemplářů znaků a_i a pokud ano, tak si označí jednu 1 mezi $\triangle$ a $|_{s_1}$. Jakmile tam všechny jedničky vyčerpá, tak došel k požadované posloupnosti. Do té doby vždy v kroku 5. načte lexikograficky následující řetězec. V kroce 4.(b) pak nahradí a_i binárními řetězci β_i a uvede pásku do kýženého pořádku.

1. T transformuje vstup do tvaru '$\ast\ (n)_2\ ,\ (p)_2\ ,\ (s_1)_2\ ,\ ...\ ,\ (s_v)_2$' (viz lemma 4.1.7)

2. T expanduje každý vstup párovací funkce jako v lemmatu 4.1.6. Jestliže při tom vzniknou nějaké prázdné symboly na konci daných podřetězců, tak se jich zbaví prostě tak, že zbytek konfiuračního řetězce posune doleva. Tzn. na konci kroku 2. je konfigurační řetězec tvaru:
   '$\ast\ 1^{(n)}\triangle\ 1^{(p)}\ |_{s_1}1^{(s_1)}\ |_{s_2}1^{(s_2)}\ |_{s_3}...\ |_{s_v}1^{(s_v)}\ @$'

3. Nechť a je první neoznačený prvek za $\ast$. Pokud
   (a) $a = 1$, tak T označí a a místo prvního ϵ na pásce vloží symbol a_1.
   (b) $a = \triangle$, pak T odznačí všechny prvky mezi $\ast$ a $\triangle$. Pokračuje krokem 4.

4. Na pásce je nyní řetězec '$\ast\ \alpha\ @\ \beta$', kde $\alpha,\beta \in A^*$. Nechť a je první neoznačený prvek řetězce '$@\ \beta$' zprava. Pokud
   (a) $a = a_i$, potom buď b první neozněčený prvek řetězece začínajícího na pravém sousedovi $|_{s_i}$. Jestliže
      i. $b = 1$, tak T označí oba prvky a,b. Pokračuje krokem 4.
      ii. $b \neq 1$, tak T odznačí všechny prvky za $\triangle$ a pokračuje krokem 5.
   (b) $a = @$, potom T označí první neoznačený prvek za '$\triangle$'. Nechť c je jeho pravý soused. Jestliže
      i. $c = |_{s_1}$, tak T všechny prvky mezi $\ast$ a @ nahradí ϵ a každý symbol a_i substituuje za řetězec β_i. Potom skončí práci.

        ii. $c = 1$, tak pokračuje krokem 5.

5. Na pásce je nyní '$* \ \alpha$ @ β'. Nechť a je první neoznačený prvek β zprava. Jestliže
    (a) $a = a_v$, pak T místo a vloží symbol $\dot{a}_1$. Pokačuje krokem 5.
    (b) $a = a_i$, $i = 1,2,...,v-1$ pak T místo a vloží symbol a_{i+1}, odznačí všechy prvky za @ a pokračuje krokem 4.

□

Nyní již máme vše potřebné k dokázání kýžené věty o asymptotickém vztahu K a I. V literatuře se obvykle připisuje Alexandru Zvonkinovi. (Zvonkin a Levin, 2007, Eq. 5.18)

Věta 4.1.9. *(Zvonkin) Nechť M_n je náhodná zpráva se zdrojovou veličinou X, potom*

$$K(M_n) \approx nH(X)$$

Důkaz. Tvrzení rozdělíme na dvě nervnosti

- Z definice Kolmogorovovské složitosti je očividné, že existuje zobrazení $\psi : \bigcup\limits_n Im(M_n) \to \{0,1\}^*$ takové, že $\ell(\psi(m)) = K(m)$ pro každou možnou zprávu $m \in Im(M_n)$ a $n \in \mathbb{N}$. Kódové slovo $\psi(m)$ je v tomto případě právě prvek $D(\mathcal{U})$ minimální možné délky, splňující $\mathcal{U}(\psi(m)) = m$, kde $\mathcal{U}$ je náš fixní stroj. Jelikož ψ je kódováním, tak dle věty 2.3.6 pro každé $\epsilon > 0$

$$\begin{aligned} P\Big(1-\epsilon \leq \frac{K(M_n)}{nH(X)}\Big) &= \\ &= P\Big(1-\epsilon \leq \frac{\ell(\psi(M_n))}{nH(X)}\Big) \xrightarrow{n\to\infty} 1 \end{aligned}$$

- Ve tvrzení 4.1.8 jsme našli stroj T, který na vstupu $\rho(m)$ pro všechna $m \in Im(M_n)$, dal výstup $T(\rho(m)) = m$. Odsud triviálně $K_T(m) \leq \ell(\rho(m))$ pro každé $m \in Im(M_n)$. Z tvrzení 3.3.2 navíc existuje $c \in \mathbb{N}$, že

$$K(m) \leq K_T(m) + c \leq \ell(\rho(m)) + c \tag{4.1}$$

  Dále si uvědomme, že z tvrzení 4.1.5 plyne $\frac{\ell(\rho(M_n))}{nH(X)} \xrightarrow{P} 1$. Ze Sluckého věty tedy $\frac{\ell(\rho(M_n))+c}{nH(X)} \xrightarrow{P} 1$. Odsud

$$\begin{aligned} 1 \geq P\Big(&\frac{K(M_n)}{nH(X)} \leq 1+\epsilon\Big) \\ &\overset{(4.1)}{\geq} P\Big(\frac{\ell(\rho(M_n) + c}{nH(X)} \leq 1+\epsilon\Big) \xrightarrow{n\to\infty} 1 \end{aligned}$$

Aplikací tvrzení 2.2.6 tedy z obou bodů dohromady dostaneme

$$P\Big(\Big|\frac{K(M_n)}{nH(X)} - 1\Big| \leq \epsilon\Big) = P\Big(1-\epsilon \leq \frac{K(M_n)}{nH(X)} \leq 1+\epsilon\Big) \xrightarrow{n\to\infty} 1$$

pro každé $\epsilon > 0$ a to jsme chtěli. □

Důsledek 4.1.10.

$$K(M_n) \approx I(M_n)$$

Důkaz. Ve tvrzení 2.2.9 jsme ukázali, že $I(M_n) \approx nH(X)$ a ve tvrzení 4.1.9, že $K(M_n) \approx nH(X)$. Z lemmatu 2.2.7 tedy plyne $K(M_n) \approx I(M_n)$ a to jsme chtěli. □

Jeden způsob, kterým lze Zvonkinovu větu interpretovat můžeme inspirovat jedním problémem z lingvistiky. Jedná se o takzvaný paradox použití/odkazování. (Chedid, 2017) Uvažme větu:

Karel má pět písmen

tato věta lze v závislost na kontextu chápat dvěma způsoby: Buď tím myslíme, že slovo Karel má pět písmen. V tomto případě podmět této věty používáme. Nebo tím myslíme, že člověk jménem Karel vlastní pět písmen. V takovém případě argument věty odkazujeme. Kolmogorovovská složitost měří informaci uvnitř řetězce, svůj argument používá, zatímco Shannonova informace řetězec ignoruje a měří informaci spojenou s výběrem zprávy z nějaké předurčené množiny. Svůj argument tedy odkazuje. Zvonkinova věta říká, že tyto dva přístupy vycházejí v pravděpodobnosti asymptoticky (skoro) stejně. Také si však z tohoto pohledu na věc lze uvědomit, že na vybraných zprávách mohou tyto dvě veličiny vycházet zcela odlišně. To si ukážeme v následujících dvou příkladech:

Příklad 4.1.3. *Uvažujme náhodnou zprávu M_n jejíž písmena tvoří náhodný výběr z veličiny X splňující $P(X = 1) = \frac{1}{2} = P(X = 0)$. Potom $I(1^{(n)}) = \log_2(2^n) = n$. V lemmatu 4.1.6 jsme dále ukázali existenci stroje T, který na všech vstupech $(n)_2, n \in \mathbb{N}$ definoval výstup $1^{(n)}$. Z věty 3.3.2 víme, že*

$$K(1^{(n)}) \leq K_T(1^{(n)}) + \mathcal{O}(1) \leq \log_2(n) + \mathcal{O}(1), \forall x \in D(T)$$

Z asymptotiky dílčích funkcí navíc existuje $n_0 \in \mathbb{N}$ splňující $\forall n \in \mathbb{N}, n \geq n_0 : \log_2(n) + \mathcal{O}(1) < n$. Odsud jsme ovšem pro $n \in \mathbb{N}$ dostatečně velké našli $x \in Im(M_n)$, že

$$K(x) << I(x)$$

Zatímco Kolmogorovovská složitost totiž měří vnitřní strukturu daného řetězce, tak Shannonova informace se o řetězec samotný vůbec nezajímá. Z jistého pohledu je Shannonova informace spíše charakteristikou zdroje než samotné zprávy. Vzhledem k tomu, že výskyt $1^{(n)}$ má stejnou pravděpodobnost jako výskyt kteréhokoliv jiného řetězce dané délky, tak je skoro jedno který řetězec nakonec budeme kódovat. Protože Shannonova informace svůj argument odkazuje, tak nemá jako koncept schopnost uchopit o jak jednoduchou posloupnost se ve skutečnosti jedná.

Příklad 4.1.4. *Z věty 3.3.4 víme, že pro každé $n \in \mathbb{N}$ existuje nekomprimovatelná binární posloupnost dané délky, označme ji x_n. Z definice nekomprimovatelné posloupnosti platí $K(x_n) \geq n$. Ať navíc X je náhodná veličina, která nabývá řetězce x_n s pstí $P(X = x_n) = \frac{1}{2}$ a nějakého jiného řetězce y_n s pstí také $\frac{1}{2}$. Pokud je pravděpodobnostní rozdělení této veličiny známé straně, která zprávu přijímá, tak lze této znalosti využít k tomu, aby se obě strany domluvili na kódování $x_n \to 1$ a $y_n \to 0$. To samozřejmě reflektuje i Shannonova informace obou* *řetězců* $I(x_n) = I(y_n) = \log_2(2) = 1$. *Uvědomme si, že podle toho o jaké* $n > 1$ *se jednalo, pak může být rozdíl obou veličin značný:*

$$I(x_n) = 1 \leq n \leq K(x_n)$$

Kolmogorovovská složitost totiž nijak nepracuje s tím, jak daný řetězec vzniknul. Svůj argument používá. Kdyby však nějaký stacionární zdroj emitoval zprávu složenou pouze z podřetězců x_n *a* y_n, *pak pravidelnost, která by se v tom řetězci pro dlouhé zprávy začala projevovat, byla jistě využitelná k výhodné kompresi. Rozdíl mezi Shannonovou informací a Kolmogorovovskou složitostí by z věty 4.1.9 pro dané zprávy s délkou klesal, až by v podstatě vymizel.*

4.2 Nerovnosti

Poměrně jednoduchým důsledkem Zvonkinovy věty je, že najdeme-li nějakou lineární nerovnost pro Kolmogorovovu složitost, potom daná nerovnost bude platit i pro Shannonovu entropii. Tento důsledek se pokusíme ukázat v této podkapitole. Nejprve, ale budeme muset nějak formálně zadefinovat, co myslíme (lineární) kolmogorovovskou a entropickou nerovností.

Značení. *Ať* $M = \{i_1,...,i_k\} \subseteq \{1,...,n\}$ *a nechť* $\bar{X} := (X_1,...,X_n)$ *je konečný náhodný vektor, potom označme* $\bar{X}_M = (X_{i_1},....,X_{i_k})$.

Definice 4.2.1. *Mějme konečný náhodný vektor* $\bar{X} = (X_1,...,X_n)$ *a reálný vektor* $\lambda = (\lambda_\emptyset,\lambda_{\{1\}},...,\lambda_{\{1,2,..,n\}}) \in \mathbb{R}^{n^2}$. *Potom řekneme, že pro* $\bar{X}$ *platí lineární entropická nerovnost s koeficienty* λ, *pokud*

$$\sum_{M\subseteq\{1,2,...,n\}} \lambda_M H(\bar{X}_M) \geq 0$$

Pokud navíc platí lineární entropická nerovnost s koeficienty λ *pro každý konečný náhodný vektor* $\bar{Y}$ *délky* n, *pak řekneme, že platí (obecný tvar) lineární entropické nerovnosti s koeficienty* λ.

Podobná definice platí i pro obecný tvar Kolmogorovovské nerovnosti. Je dobré si uvědomit, že zde do daných nerovností vstupuje náhodná veličina. Vzpomeňme si, že nerovnost obsahující náhodnou veličinu z definice 2.2.2 platí, pokud je splněna s pravděpodobností rovnou 1.

Definice 4.2.2. *Mějme konečný náhodný vektor* $\bar{B} = (B_1,...,B_n)$, *kde náhodné veličiny* B_i *splňují* $Im(B_i) \subseteq \{0,1\}^*$ *a reálný vektor* $\lambda = (\lambda_\emptyset,\lambda_{\{1\}},...,\lambda_{\{1,2,..,n\}})$ *z* $\mathbb{R}^{n^2}$. *Potom řekneme, že pro* $\bar{B}$ *platí lineární Kolmogorovovská nerovnost s koeficienty* λ, *pokud*

$$\sum_{M\subseteq\{1,2,...,n\}} \lambda_M K(\bar{B}_M) \geq 0$$

Pokud navíc platí lineární Kolmogorovovská nerovnost s koeficienty λ *pro každý konečný náhodný vektor* $\bar{D} = (D_1,D_2,...,D_n)$ *splňující* $Im(D_i) \subseteq \{0,1\}^*$, *pak řekneme, že platí (obecný tvar) lineární Kolmogorovovské nerovnosti s koeficienty* λ.

Lemma 4.2.1. *Nechť* $\bar{X} = (X_1,....,X_n)$ *je konečný náhodný vektor,* R *nějaká končená množina a* $\psi : Im(X) \to R$ *prosté zobrazení. Potom*

$$H(\bar{X}) = H(\psi(\bar{X}))$$

Důkaz. Protože existuje jednoznačná korespondence mezi $Im(\psi(\bar{X}))$ a $Im(\bar{X})$, tak je tvrzení v podstatě jednoduchým důsledkem definice entropie:

$$\begin{aligned} H(\psi(\bar{X})) &= \sum_{a\in Im(\psi(\bar{X}))} -P(\psi(\bar{X}) = a)\log_2(P(\psi(\bar{X}) = a)) \\ &= \sum_{a\in Im(\psi(\bar{X}))} -P(\bar{X} = \psi^{-1}(a))\log_2(P(\bar{X} = \psi^{-1}(a))) \\ &= \sum_{b\in Im(\bar{X})} -P(\bar{X} = b)\log_2(P(\bar{X} = b)) \end{aligned}$$

□

Nyní už máme vše potřebné k dokázání věty o nerovnostech, ta se poprvé objevila v článku (Hammer a kol., 2000, Theorem 1).

Věta 4.2.2. *(Hammer) Nechť platí lineární Kolmogorovovská nerovnost s koeficienty* $\lambda = (\lambda_{\emptyset},\lambda_{\{1\}},...,\lambda_{\{1,2,...,n\}}) \in \mathbb{R}^{2^n}$*. Potom platí lineární entropická nerovnost s koeficienty* λ*.*

Důkaz. Nechť $\bar{X} = (X_1,X_2...,X_n)$ je konečný náhodný vektor. Navíc uvažujme libovolná prostá zobrazení $\phi_i : Im(X_i) \to \{0,1\}^*$ a náhodné veličiny $B_i := \phi(X_i)$. Potom buď $M = \{i_1,i_2,...,i_k\}$ libovolná neprázdná podmnožina $\{1,2,..,n\}$. Párovací funkce $<\ ,\ >$ nám umožňuje jednoznačně sloučit $B_i, i \in M$ do jedné náhodné veličiny $\beta_M :=< B_{i_1},...,B_{i_k} >$. Nyní mějme náhodný výběr $\beta_M^{(1)},\beta_M^{(2)},...,\beta_M^{(N)}$ z β_M, pro libovolné $N \in \mathbb{N}$ (existence náhodného výběru je popsána například v (Kallenberg, 2002, Theorem 2.19)). Potom z věty 4.1.9

$$\frac{K(\beta_M^{(1)},\beta_M^{(2)},...,\beta_M^{(N)})}{N} \xrightarrow{P} H(\beta_M)$$

Ale $<\ ,\ >$: $(\{0,1\}^*)^n \to (\{0,1\})^*$ a $\phi = (\phi_1,\phi_2,...,\phi_n)$ jsou prostá zobrazení. Tedy z lemmatu 4.2.1 ihned:

$$H(\beta_M) = H(< B_{i_1},B_{i_2},...,B_{i_k} >) = H(B_{i_1},B_{i_2},....,B_{i_k}) = H(\bar{X}_M)$$

Takže jsme pro libovolné neprázdné $M \subseteq \{1,2,...,n\}$ dokázali:

$$\frac{K(\beta_M^{(1)},\beta_M^{(2)},...,\beta_M^{(N)})}{N} \xrightarrow{P} H(\beta_M) = H(\bar{X}_M)$$

Jelikož ale $< \beta_M^{(1)},..,\beta_M^{(N)} >$ je pro každé $M \subseteq \{1,2,...,n\}$ konečná binární náhodná veličina a z předpokladu platí obecný tvar lineární Kolmogorovovské nerovnosti s koeficienty λ, tak platí speciálně i pro všechny veličiny $< \beta_M^{(1)},..,\beta_M^{(N)} >$, kde $M \subseteq \{1,2,...,n\}$. Dohromady se Sluckého větou tedy:

$$0 \le \sum_{M\subseteq\{1,2,...,n\}} \lambda_M \frac{K(\beta_M^{(1)},\beta_M^{(2)},...,\beta_M^{(N)})}{N} \xrightarrow{P} \sum_{M\subseteq\{1,2,...,n\}} \lambda_M H(\bar{X}_M) \qquad (4.2)$$

Buď $\epsilon > 0$, tvrdím, že $P(\sum_{M\subseteq\{1,2,...,n\}} \lambda_M H(\bar{X}_M) > -\epsilon) = 1$. Pro usnadnění notace označme $H := \sum_{M\subseteq\{1,2,...,n\}} \lambda_M H(\bar{X}_M)$ a $K^{(N)} := \sum_{M\subseteq\{1,2,...,n\}} \lambda_M K(\beta_M^{(1)},...,\beta_M^{(N)})$

Neboť z předpokladu $P(K^{(N)} \ge 0) = 1$ a z (4.2) $P(H > K^{(N)} - \epsilon) \xrightarrow{P} 1$, tak z lemmatu 2.2.6:

$$P(H > -\epsilon) \ge P(H > K^{(N)} - \epsilon \cap K^{(N)} \ge 0) \xrightarrow{N\to\infty} 1$$

Tedy $P(H > -\epsilon) \xrightarrow{N\to\infty} 1$ pro libovolné $\epsilon > 0$. Ale H nezávisí na N, tedy $P(H > -\epsilon) = 1, \forall\epsilon > 0$. Odsud $P(H \ge 0) = 1$, pro H konstantní a konečně

$$\sum_{M\subseteq\{1,2,...,n\}} \lambda_M H(\bar{X}_M) \ge 0$$

□

Závěr

Cílem této práce bylo formálně zavést a porovnat dvě formalizace kvantitativního aspektu informace uložené ve zprávě: Kolmogorovovskou složitost a Shannonovu informaci.

Ve druhé kapitole jsme zavedli Shannonovu informaci jakožto funkci v pravděpodobnosti. Stála za ní intuitivní myšlenka, že s nastáním pravděpodobného jevu se pojí méně informace než s jevem nepravděpodobným. Na základě jistých „rozumných“ vlastností jsme ukázali, že taková funkce je v podstatě určena jenoznačně. V průběhu této práce jsme si navíc uvědomili, že pokud přijmeme nějaké základní předpoklady z teorie pravděpodobnosti, potom nám Shannonova informace dává spodní odhad na délku binární komprese příslušné zprávy. Formálně to znamená, že vezmeme-li jakékoliv binární kódování všech možných zpráv nad určitou abecedou, potom pravděpodobnost toho, že kódové slovo libovolně zvolené zprávy bude výrazně kratší než je Shannonova informace dané zprávy, jde s rostoucí délkou zpráv k nule. Navíc jsme našli binární kódování, které se této hranici bylo schopno až na zanedbatelnou chybu asymptoticky přiblížit.

Na začátku třetí kapitoly jsme ovšem vysvětlili, že Shannnova informace má pro aplikovatelnost na binární kompresi zpráv v reálném světě určité rezervy. Jako rozumnou alternativu jsme tedy uvedli Kolmogorovovskou složitost. Ta je definovná jako nejkratší algoritmický popis příslušné zprávy. Kolmogorov jako formalizaci algoritmického popisu zprávy přijal délku minimálního vstupu do univerzálního Turingova stroje, na kterém tuto zprávu emituje. Díky vlastnostem univerzálních strojů se ukázalo, že Kolmogorovovská složitost až na konstantní člen nezávisí na výběru univerzálního Turingova stroje. Narazili jsme ovšem na to, že neexistuje algoritmus, který by Kolmogorovovskou složitost spočetl.

Obě definice mají tedy své zející problémy. Kolmogorovovská složitost nejde spočítat, ale je dobře filosoficky ukotvena a Shannonova informace jde snadno odhadnout, ale jsou s ní spojeny problémy aplikovatelnosti v praxi. Ukázali jsme však, že mezi těmito dvěmi komplementárními definicemi existuje most. Ten jsme pojmenovali Zvonkinova věta. Ta říká, že přijmeme-li předpoklady Shannonovy teorie, potom vychází Kolmogorovovská složitost a Shannonova informace až na zanedbatelnou chybu asymptoticky stejně. Tato věta nám jinými slovy sděluje, že algoritmický popis zprávy a míra „statistické“ informace uvolněné s posláním této zprávy vycházejí přibližně stejně. V důsledku toho jsme poměrně snadno ukázali Hammerovu větu uvádějící, že pokud platí nějaká lineární nerovnost pro Kolmogorovovskou složitost, potom platí i pro Shannonovu entropii.

Seznam použité literatury

Adriaans, P. (2019). Information. `https://plato.stanford.edu/archives/spr2019/entries/information/`.

Aftab, O., Cheung, P., Kim, A., Thakkar, S. a Yeddanapudi, N. (2001). Information theory and the digital age. *The Structure of Engineering Revolutions, Massachusetts Institute of Technology.* URL `http://web.mit.edu/6.933/www/Fall2001/Shannon2.pdf`.

Avigad, J. The birth of model theory: L owenheim's theorem in the frame of thetheory of relativesplato's cave analysis. URL `https://www.andrew.cmu.edu/user/avigad/Reviews/badesa_review.pdf`.

Bennett, C. (2003). Logical depth and physical complexity. doi: 10.1007/978-3-7091-6597-3_8.

Bezhanishvili, G. a Moss, L. (2019). Undecidability of first-order logic. URL `https://www.cs.nmsu.edu/historical-projects/Projects/FoLundecidability.pdf`.

Burgin, M. (2018). Ideas of plato in the context of contemporary science and mathematics. *Athens Journal of Humanities and Arts*, **4**. doi: 10.30958/ajha.4.3.1.

Carter, T. An introduction to information theory and entropy. URL `https://www.csustan.edu/sites/default/files/CS/Carter/documents/info-lec.pdf`.

Chedid, F. (2017). Kolmogorov complexity and information content. URL `https://pdfs.semanticscholar.org/107c/67bd106686312aa516dd4b33c38daf7b0f61.pdf`.

Cover, T. M. a Thomas, J. A. (2006). *Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing).* Druhé opravené vydání. Wiley-Interscience New York, NY, USA. ISBN 0471241954.

Durand, B. a Zvonkin, A. (2007). Kolmogorov complexity. In Charpentier, E., LESNE, A. a Nikolski, N. K., editors, *Kolmogorov's Heritage in Mathematics*, pages 281–291. ISBN 3540363491.

Gleick, J. (2011). *The Information A History, a Theory, a Flood.* Knopf Doubleday Publishing Group. ISBN 0375423729.

Grünwald, Peter a Vitányi, P. (2003). Kolmogorov complexity and information theory. with an interpretation in terms of questions and answers. *J. Logic Lang. Inform*, **12**, 497–529. doi: 10.1023/A:1025011119492.

Hamilton, E. a Cairns, H. (1992). *The Collected Dialogues of Plato.* Princeton University Press. ISBN 9780691097183.

Hammer, D., Romashchenko, A., Shen, A. a Vereshchagin, N. (2000). Inequalities for shannon entropy and kolmogorov complexity. *Journal of Computer and System Sciences*, **60**, 442–464. doi: 10.1006/jcss.1999.1677.

Hartley, R. V. L. (1928). Transmission of information. *Bell System Technical Journal*, **7**, 553–563. doi: 10.1002/j.1538-7305.1928.tb01236.x.

Hilbert, D. a Ackermann, W. F. (1999). *Principles of Mathematical Logic*. Anglický překlad druhého přepracovaného vydání. Matfyzpress. ISBN 9780821820247.

Hlubinka, D. (2018). Nmai059 pravděpodobnost a statistika příručka k přednášce. URL `http://www.karlin.mff.cuni.cz/~hlubinka/soubory/nmai059_skripta_2019.pdf`.

Jones, G. A. a Jones, J. M. (2000). *Information and Coding Theory*. Springer London. ISBN 1852336226.

Kallenberg, O. (2002). *Foundations of Modern Probability (Probability and Its Applications)*. Druhé přepracované vydání. Springer New York. ISBN 9780387953137.

Kolmogorov, A. (1968). Three approaches to the quantitative definition of information. *International Journal of Computer Mathematics*, **2**, 157–168. doi: 10.1080/00207166808803030?journalCode=gcom20#.VGfIavmsVFo.

Kolmogorov, A. N. (1998). On tables of random numbers. ISSN 0304-3975. URL `http://dx.doi.org/10.1016/S0304-3975(98)00075-9`.

Lambalgen, M. (2002). Randomness and foundations of probability: Von mises' axiomatisation of random sequences. doi: 10.1214/lnms/1215453582.

Li, M. a Vitányi, P. (2008). *An Introduction to Kolmogorov Complexity and Its Applications (Texts in Computer Science)*. Třetí přepracované vydání. Springer New York. ISBN 9780387339986.

Logan, R. (2012). What is information?: Why is it relativistic and what is its relationship to materiality, meaning and organization. *Information*, **33390**, 68–91. doi: 10.3390/info3010068.

McLennan, S. (2011). Human computers. URL `https://crgis.ndc.nasa.gov/historic/Human_Computers`.

Nau, R. (2001). De finetti was right: Probability does not exist. *Theory and Decision*, **51**, 89–124. doi: 10.1023/A:1015525808214.

Oxford English Dictionary Contributors (2009). information, n. URL `https://www.oed.com/viewdictionaryentry/Entry/95568?print`.

Porter, C. (2014). Kolmogorov on the role of randomness in probability theory. *Mathematical Structures in Computer Science*, **24**. doi: 10.1017/S0960129512000801.

SHAGRIR, O. (2007). Gödel on turing on computability. In OLSZEWSKI, A., WOLE'NSKI, J. a JANUSZ, R., editors, *Church's Thesis After Seventy Years. Ontos Verlag*, pages 393–419. ISBN 3938793090.

SHANNON, C. (1948). A mathematical theory of communication. *Bell System Technical Journal*, **27**, 379–423. doi: 10.1145/584091.584093.

SHANNON, C. (1956). The bandwagon. *IRE Transactions on Information Theory*, **2**(1), 3–3. doi: 10.1109/TIT.1956.1056774.

SHANNON, C. E. (1950). The lattice theory of information. In JACKSON, W., editor, *Symposium on Information Theory (Report of Proceedings)*, pages 105–107. The Ministry of Supply. Konalo se v Lecture Theatre of the Royal Society, Burlington House, September 1950.

SIPSER, M. (2012). *Introduction to the Theory of Computation.* Třetí opravené vydání. Cengage Learning. ISBN 9781133187813.

TEUTSCH, J. Universal turing machine. URL `http://people.cs.uchicago.edu/~simon/OLD/COURSES/CS311/UTM.pdf`.

THE EDITORS OF ENCYCLOPAEDIA BRITANNICA (2011). Form. URL `https://www.britannica.com/topic/form-philosophy`.

TURING, A. M. (1937). On computable numbers, with an application to the entscheidungsproblem. URL `https://londmathsoc.onlinelibrary.wiley.com/doi/abs/10.1112/plms/s2-42.1.230`.

WIKIPEDIA CONTRIBUTORS (2019a). Information age — Wikipedia, the free encyclopedia. URL `https://en.wikipedia.org/w/index.php?title=Information_Age&oldid=923414177`. [Online; accessed 28-October-2019].

WIKIPEDIA CONTRIBUTORS (2019b). Shannon's source coding theorem — Wikipedia, the free encyclopedia. URL `https://en.wikipedia.org/w/index.php?title=Shannon%27s_source_coding_theorem&oldid=899781871`. [Online; accessed 28-October-2019].

WILSON, T. (2011). Plato's cave analysis. URL `https://www.youtube.com/watch?v=axARKd24eHo`.

WOLF, R. S. (2005). *A Tour through Mathematical Logic.* The Mathematical Association of America. ISBN 9780883850367.

YEUNG, R. W. (2008). *INFORMATION THEORY AND NETWORK CODING.* Springer. ISBN 0387792333.

ZACH, R. (2006). *Hilbert's Program Then and Now*, pages 411–447. doi: 10.1016/B978-044451541-4/50014-2.

ZVONKIN, A. a LEVIN, L. (2007). 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*, **25**, 83. doi: 10.1070/RM1970v025n06ABEH001269.

Švejdar, V. (2002). *Logika neúplnost, složitost a nutnost.* ACADEMIA. ISBN 802001005X.