FAKULTA INFORMAČNÍCH TECHNOLOGIÍ ČVUT V PRAZE

Zadání bakalářské práce

Název:	Algoritmy pro výpočet Ludolphova čísla a ověření normality jeho aproximací
Student:	Martin Kostrubanič
Vedoucí:	doc. Ing. Ivan Šimeček, Ph.D.
Studijní program:	Informatika
Obor / specializace:	Teoretická informatika
Katedra:	Katedra teoretické informatiky
Platnost zadání:	do konce letního semestru 2021/2022

Pokyny pro vypracování

1) Nastudujte a popište Ludolphovo číslo, jeho vlastnosti, historii a využití ve vědě.
2) Popište algoritmy(zejména Chudnovskyho algoritmus [1]) pro výpočet Ludolphova čísla využívající nekonečné řady. Stručně popište i iterační algoritmy a algoritmus BBP [2].
3) Pomocí knihovny GNU MPFR implementujte algoritmy z bodu 2) a porovnejte rychlost jejich konvergence.
4) Ověřte, že různě přesné aproximace Ludolphova čísla jsou normální čísla.

–

[1] https://www.craig-wood.com/nick/articles/pi-chudnovsky/[2] https://www.experimentalmath.info/bbp-codes/bbp-alg.pdf

---

Elektronicky schválil/a doc. Ing. Jan Janoušek, Ph.D. dne 10. září 2020 v Praze.

FAKULTA
INFORMAČNÍCH
TECHNOLOGIÍ
ČVUT V PRAZE

Bakalářská práce

Algoritmy pro výpočet Ludolphova čísla a ověření normality jeho aproximací

Martin Kostrubanič

Katedra Teoretické Informatiky
Vedoucí práce: doc. Ing. Ivan Šimeček, Ph.D.

2. května 2021

Poděkování

Chtěl bych poděkovat vedoucímu práce za jeho rady a připomínky a také rodičům a přátelům, kteří mě podporovali.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů. V souladu s ust. § 2373 odst. 2 zákona č. 89/2012 Sb., občanský zákoník, ve znění pozdějších předpisů, tímto uděluji nevýhradní oprávnění (licenci) k užití této mojí práce, a to včetně všech počítačových programů, jež jsou její součástí či přílohou a veškeré jejich dokumentace (dále souhrnně jen „Dílo"), a to všem osobám, které si přejí Dílo užít. Tyto osoby jsou oprávněny Dílo užít jakýmkoli způsobem, který nesnižuje hodnotu Díla a za jakýmkoli účelem (včetně užití k výdělečným účelům). Toto oprávnění je časově, teritoriálně i množstevně neomezené. Každá osoba, která využije výše uvedenou licenci, se však zavazuje udělit ke každému dílu, které vznikne (byť jen zčásti) na základě Díla, úpravou Díla, spojením Díla s jiným dílem, zařazením Díla do díla souborného či zpracováním Díla (včetně překladu) licenci alespoň ve výše uvedeném rozsahu a zároveň zpřístupnit zdrojový kód takového díla alespoň srovnatelným způsobem a ve srovnatelném rozsahu, jako je zpřístupněn zdrojový kód Díla.

V Praze dne 2. května 2021 . . . . . .

České vysoké učení technické v Praze
Fakulta informačních technologií
© 2021 Martin Kostrubanič. Všechna práva vyhrazena.
Tato práce vznikla jako školní dílo na Českém vysokém učení technickém v Praze, Fakultě informačních technologií. Práce je chráněna právními předpisy a mezinárodními úmluvami o právu autorském a právech souvisejících s právem autorským. K jejímu užití, s výjimkou bezúplatných zákonných licencí a nad rámec oprávnění uvedených v Prohlášení na předchozí straně, je nezbytný souhlas autora.

Odkaz na tuto práci

Kostrubanič, Martin. *Algoritmy pro výpočet Ludolphova čísla a ověření normality jeho aproximací.* Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2021.

Abstrakt

Práce se zabývá Ludolfovým číslem, algoritmy pro jeho výpočet a jeho normalitou. V teoretické části je popsáno Ludolfovo číslo a vybrané algoritmy pro jeho aproximaci. V praktické části jsou tyto algoritmy implementovány pomocí knihovny GNU-MPFR. Dále je porovnána rychlost jejich konvergence. V poslední části práce je zkoumáno, jestli jsou aproximace Pí normální čísla. K tomuto účelu je porovnán výskyt jednotlivých číslic a řetězců délky 2 v desítkové a šestnáctkové bázi.

Při testování rychlosti konvergence vycházejí nejlépe Chudnovskyho algoritmus a Gauss-Legendreho algoritmus. Na základě výsledků získaných při vyšetřování normality je možné říci, že zkoumané aproximace jsou normální čísla.

Klíčová slova Ludolfovo číslo, aproximační algoritmus, rychlost konvergence, C++, GNU-MPFR, normalita čísla

Abstract

Thesis deals with Ludolphian number, algorithms for calculating its value and its normality. Ludolphian number and chosen approximation algorithms are described in the theoretical part. These algorithms are implemented with

GNU MPFR library in the practical part. The speed of convergence of those algorithms is also measured. At the end, the normality of approximations of Pi is tested. The occurrence of single digits and strings of length 2 in base 10 and 16 is compared for this purpose.

The biggest speed of convergence was measured by Chudnovsky algorithm and Gauss–Legendre algorithm. By results achieved from normality testing it is safe to say, that tested approximations of Pi are normal numbers.

Keywords Ludolphian number, approximation algorithm, speed of convergence, C++, GNU-MPFR, normality of a number

Obsah

Seznam obrázků

Seznam tabulek

Úvod

Ludolfovo číslo je nejdůležitější konstanta v matematice. Jeho využití však oblast matematiky značně přesahuje a najdeme ho i v mnoha rovnicích fyziky. Ačkoliv pro praktické využití stačí číslo Pí aproximovat na několik desítek desetinných míst, stále padají nové rekordy v počtu vyčíslených cifer. K dosažení takových výsledků je třeba velmi vysoký výpočetní výkon, ale také korektní a efektivní algoritmus. Algoritmů pro počítání Ludolfova čísla je mnoho a není vyloučeno, že budou vznikat další.

Práce je určena pro všechny, které jako mě zajímá matematika, a zejména Ludolfovo číslo. Práce nabízí popis nejdůležitějších algoritmů, jejich implementaci a porovnání jejich rychlosti na jednom místě.

Cílem práce je nastudovat a popsat Ludolfovo číslo a algoritmy pro jeho počítání. Dalším cílem je implementace algoritmů a porovnání rychlosti jejich konvergence. V neposlední řadě je cílem také určit, zda jsou aproximace Pí normální čísla.

Cílem teoretické části je popsat Ludolfovo číslo, jeho vlastnosti, historii a využití ve vědě. Dále pak nastudovat a popsat algoritmy pro výpočet čísla Pí, a to zejména ty, které využívají nekonečné řady. Také je cílem popsat ostatní algoritmy, jako jsou iterativní algoritmy nebo spigot algoritmy.

Cílem praktické části je seznámit se s knihovnou GNU-MPFR pro práci s desetinnými čísly s neomezenou přesností. Dále algoritmy popsané v teoretické části pomocí této knihovny naimplementovat, změřit rychlost jejich konvergence a porovnat výsledky. Dále je cílem zjistit, jestli jsou aproximace Ludolfova čísla normální čísla a to pro různé báze.

Kapitola 1

Definice pojmů a podobné práce

1.1 Definice pojmů

Iracionální číslo Iracionální číslo je takové, které nelze vyjádřit podílem dvou celých čísel. Znamená to, že se nedá zapsat jako číslo s ukončeným desetinným rozvojem ani jako číslo periodické, v jeho rozvoji tedy neexistuje žádný stále se opakující vzor.

Transcendentní číslo Číslo c je transcendentní, právě když neexistuje racionální polynom, jehož kořenem je c. Číslo c tedy nemůže být zapsáno jako kombinace racionálních čísel a n-tých odmocnin.

Normální číslo Číslo je bráno jako normální v bázi b, pokud každá z b cifer je v čísle zastoupena rovnoměrně s hustotou $\frac{1}{b}$ a obecně pokud pro všechna přirozená n, pro všechny možné řetězce číslic délky n platí, že se v čísle vyskytují rovnoměrně s hustotou b^{-n}.

Absolutně normální číslo Číslo je absolutně normální, pokud je normální pro všechny báze větší nebo rovny dvěma.

Nekonečná řada Nekonečná řada je posloupnost částečných součtů posloupnosti $(a_n)_{n=1}^{\infty}$. Značíme: $\sum_{n=1}^{\infty} a_n$.

Řád konvergence Aproximační algoritmus je řádu konvergence $n(r = n)$, právě když se výsledná aproximace při každé iteraci n-násobně přiblíží ke skutečné hodnotě.

1.2 Práce na podobné téma

Podobným tématem se již zabývalo množství autorů. Například Madelaine Jansson se ve své bakalářské práci Approximation of Pi zabýval algoritmy pro výpočet Pí. Zabýval se ale pouze několika málo algoritmy a na rozdíl od této práce neporovnával rychlost jejich konvergence. Henrik Vestermark ve své práci Practical implementation of π Algorithms provedl praktickou implementaci algoritmů. Teorií okolo těchto algoritmů se ale zabýval pouze okrajově. Velmi podobným tématem se zabýval Adam Pavlis ve své bakalářské práci Výpočet Ludolfova čísla a ověření jeho vlastností. Narozdíl ode mě se více zabýval analýzou knihoven pro práci s desetinnými čísly s libovolnou přesností. Já se více zaměřil na teoretický popis algoritmů, zejména těch, využívajících nekonečné řady.

Kapitola 2

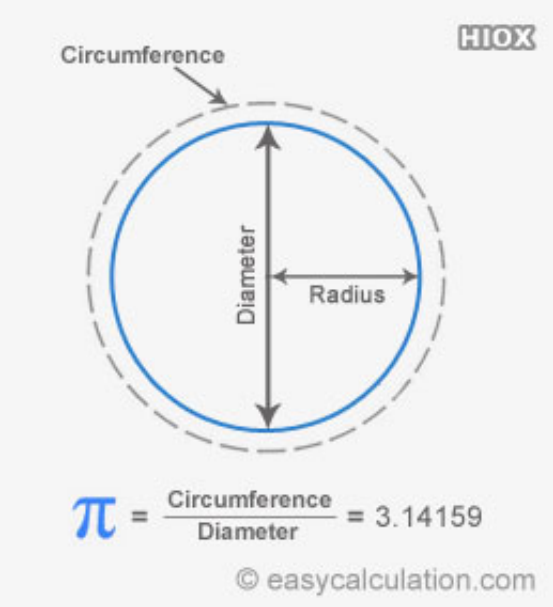
Popis Ludolfova čísla, jeho vlastností, historie a využití ve vědě

2.1 Popis

Číslo Pí(π) je matematická konstanta, nejčastěji definovaná jako poměr obvodu ku průměru kruhu. Tento poměr zůstává stejný pro jakoukoli velikost kruhu. Existuje celá řada dalších definic tohoto čísla, například pokud bychom měli kruh a čtverec o straně jeho poloměru, poměr obsahu kruhu ku obsahu čtverce by byl roven číslu Pí. Jeho název je odvozen od prvního písmena řeckého slova *perimetros*, které znamená obvod. Také je nazýváno Ludolphovo číslo podle německého matematika Ludolpha van Ceulena, který jej na přelomu 16. a 17. století vyčíslil na 35 desetinných míst.[9, strany 182–183] Můžeme se s tímto číslem setkat i jako s Archimédovou konstantou podle řeckého matematika a fyzika Archiméda ze Syrakus, který se jako první zabýval jeho výpočtem.[10]

Hodnota Ludolphova čísla je přibližně 3,1415. Honba za co nejpřesnějším vyčíslením Archimedovy konstanty začala už u starověkých civilizací. Archimedes vytvořil první algoritmus pro aproximaci Pí již ve 3. století př. n. l. V 5. století n. l. vyčíslili čínští a indičtí matematici Ludolphovo číslo na 7, respektive 5 desetinných míst. První algoritmus na aproximaci π používající nekonečné řady byl založen na Leibnizově řadě objevené ve 14. století. Díky výkonným počítačům dnešní doby a rychle konvergujícím algoritmům dokážeme dnes Ludolphovo číslo aproximovat na biliony desetinných míst. Pro praktické využití nám stačí vyčíslení na několik desítek desetinných míst. Takto přesné aproximace jsou tedy spíše o překonávání rekordů, ukázce schopností moderní techniky a v neposlední řadě o touze dozvědět se o této konstantě co nejvíce.[9, strana 17]

Číslo Pí se, vzhledem k jeho definici, pochopitelně vyskytuje v mnoha rovnicích týkajících se geometrie a trigonometrie. Objevuje se však i v méně očekávaných odvětvích matematiky jako třeba v teorii čísel nebo ve statistice. Je hojně používáno i v mnoha odvětvích fyziky. Ludolphovo číslo je známo i mimo oblast matematiky a fyziky, lidé si například snaží zapamatovat desetitisíce číslic jeho desetinného rozvoje.



Obrázek 2.1: Definice [1]

2.2 Vlastnosti

Ačkoli bývá Pí někdy aproximováno zlomkem $\frac{22}{7}$, je iracionální. Existuje několik důkazů iracionality Ludolphova čísla. Důkazy většinou používají kalkulus a důkaz sporem. Stupeň iracionality, tedy jak přesně lze číslo aproximovat pomocí zlomku, není přesně znám. Podle odhadů je ale stupeň iracionality Pí vyšší než např. u čísla e nebo $\ln 2$.[11, strany 570–572] Pí je také transcendentní číslo.[12] Důsledkem je, že Pí nelze euklidovsky zkonstruovat kružítkem a pravítkem. To znamená, že nelze provést kvadraturu kruhu, tedy zkonstruovat čtverec, jehož obsah je stejný, jako obsah kvadraturovaného kruhu. Tento problém pochází ze starého Řecka. I když je matematicky dokázáno, že kvadratura kruhu je nemožná, někteří se o to stále pokoušejí.

Normalita Ludolphova čísla nebyla ani dokázána ani vyvrácena. Výzkumy naznačují, že by Pí mohlo být normální číslo. Nicméně vyskytují se v něm sekvence číslic, které se zdají být ne úplně náhodné, jako např. sekvence šesti devítek začínající na 762. pozici desetinného rozvoje. Toto místo se nazývá

Feynmanův bod. Takováto sekvence se zdá dost podezřelá, pravděpodobnost jejího výskytu je však nenulová a normalitu rozhodně nevyvracuje.[13] Pí se, jako každé iracionální číslo, dá reprezentovat pomocí nekonečného řetězového zlomku:

$$3 + \cfrac{1}{7 + \cfrac{1}{15 + \cfrac{1}{1 + \cfrac{1}{292 + \cfrac{1}{\ddots}}}}}$$

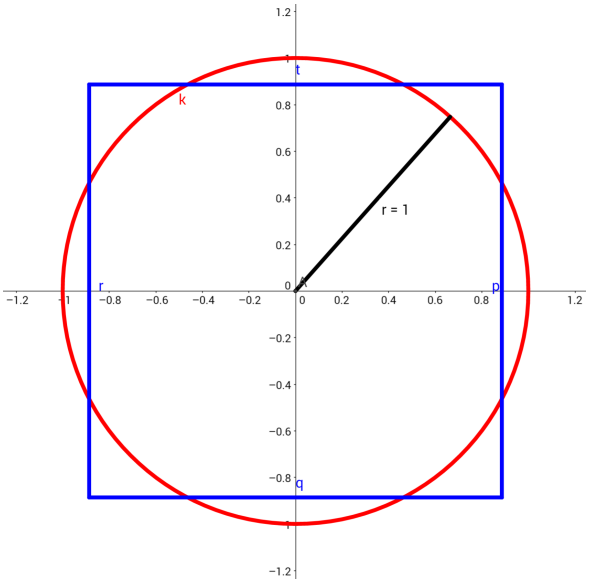
Useknutí tohoto zlomku dá vždy nějakou aproximaci Pí, např. 3, $\frac{22}{7}$ a $\frac{333}{106}$. Aproximace vzniklé tímto způsobem jsou nejlepší možné, neexistuje tedy zlomek se stejným nebo menším jmenovatelem, který by byl blíže π.[14, strana 78] Zlomek se dá generovat následujícím algoritmem [15, strany 151–152]:

$$a_0 = \lfloor \pi \rfloor = 3$$

$$u_1 = \frac{1}{\pi - 3} \approx 7,0625$$

$$a_n = \lfloor u_n \rfloor$$

$$u_{n+1} = \frac{1}{u_n - a_n}.$$



Obrázek 2.2: Kvadratura kruhu [2]

2.3 Historie

Někteří egyptologové tvrdí, že staří Egypťané používali zlomek $\frac{22}{7}$ jako aproximaci Pí již ve 3. tisíciletí př. n. l.[16] Tato tvrzení jsou však pouze dohady a jsou brána dost skepticky. Nejstarší písemný odhad Ludolphova čísla pochází od Babyloňanů z 2. tisíciletí př. n. l. jako 3,125.[9, strana 167] Indické astronomické výpočty ze 4. století př. n. l. používaly aproximaci Pí jako $\frac{339}{108}$, tedy přibližně 3,139.[17, strana 133] První algoritmus pro výpočet hodnoty Ludolphova čísla používal opsané a vepsané polygony. Vymyslel ho Archimedes asi v polovině 3. století př. n. l. Archimedes pomocí polygonů s 96 stranami určil, že $\frac{223}{71} < \pi < \frac{22}{7}$, tedy že $3.1408 < \pi < 3.1429$.[9, strana 170] V roce 265 použil čínský matematik Liu Hui algoritmus s polygony s 3072 stranami a získal hodnotu 3,1416. Asi o 200 let později použil stejný algoritmus Zu Chongzhi s 12288strannými polygony a dosáhl aproximace $3.1415926 < \pi < 3.1415927$.[18, strana 202] Další rekordy přišly až o 800 let později. V roce 1424 perský astronom Jamshid al-Kashi aproximoval Pí na 16 desetinných míst pomocí polygonů s $3 \cdot 2^{28}$ stranami.[19, strany 64–85] Ludolph van Ceulen, holandský matematik, v roce 1596 vyčíslil π na 20, později na 35 desetinných míst.[9, strany 182–183] Pomocí polygonů s 10^{40} stranami aproximoval Pí v roce 1630 rakouský astronom Christoph Grienberger na 38 desetinných míst, což je nejlepší výsledek dosažený pomocí algoritmu používající polygony.[9, strana 183] S objevem nekonečných řad, došlo k novým rekordům v aproximování Pí. První nekonečné řady k tomuto účelu vznikaly v Indii. Indický matematik Madhava ze Sangamagramy použil tyto řady na počátku 15. století k aproximaci Pí na 11 desetinných míst.[5] První nekonečná řada obsahující π objevená v Evropě, byla překvapivě nekonečný produkt:

$$\frac{2}{\pi} = \frac{\sqrt{2}}{2} \cdot \frac{\sqrt{2+\sqrt{2}}}{2} \cdot \frac{\sqrt{2+\sqrt{2+\sqrt{2}}}}{2} \cdots .$$

Přišel na ni francouzský matematik François Viète v roce 1593.[9, strana 187] Kolem roku 1660 objevili Isaac Newton a Gottfried Wilhelm Leibniz kalkulus, což vedlo ke vzniku mnoha nekonečných řad, které se daly použít k aproximaci Ludolphova čísla. Nekonečná řada:

$$\arctan z = z - \frac{z^3}{3} + \frac{z^5}{5} - \frac{z^7}{7} \cdots$$

byla objevena v roce 1671 Jamesem Gregorym a poté v roce 1674 Gottfriedem Leibnizem. Tato řada, říká se jí Madhavova nebo Gregory-Leibnizova řada, je rovna $\frac{\pi}{4}$ pro $z = 1$.[14, strany 53–54] Tuto řadu použil v roce 1699 anglický matematik Abraham Sharp s $z = \frac{1}{\sqrt{3}}$ a aproximoval tak Pí na 71 desetinných míst.[9, strana 189] V roce 1706 vytvořil John Machin podle Gregory-Leibnizovy řady jinou řadu, která konverguje mnohem rychleji [9,

strany 192–193]:

$$\frac{\pi}{4} = 4 \arctan \frac{1}{5} - \arctan \frac{1}{239}.$$

S touto řadou aproximoval Machin Pí na 100 desetinných míst. Později vznikaly různé varianty této řady. Tzv. Machin-like řady. Daniel Ferguson použil v roce 1946 jednu z nich k vyčíslení Ludolphova čísla na 620 desetinných míst.[9, strany 72–74]

Rozvoj počítačů ve 20. století znamenal obrovský pokrok v aproximaci Pí. Pomocí Machin-like řad docházelo k opakovanému překonávání rekordu. V roce 1973 bylo Pí vyčísleno na milion desetinných míst.[9, strana 197] Kolem roku 1980 vznikaly iterativní algoritmy, které byly mnohem rychlejší než doposud používané nekonečné řady, jsou ovšem mnohem paměťově náročnější.[20] Také byly objeveny algoritmy pro rychlejší násobení velkých čísel, jako např. Karacubův algoritmus.[9, strany 15–17] V osmdesátých a devadesátých letech 20. století byly objeveny nové nekonečné řady, které jsou stejně rychlé, možná i rychlejší než iterativní algoritmy, a hlavně jsou mnohem méně náročné na paměť.[20] Jedna z nich je nekonečná řada Srinivasa Ramanujana:

$$\frac{1}{\pi} = \frac{2\sqrt{2}}{9801} \sum_{k=0}^{\infty} \frac{(4k)!(1103 + 26390k)}{k!^4 (396^{4k})}.$$

[9, strany 103–104] V roce 1985 ji Bill Gosper použil na aproximaci Pí na 17 milionů desetinných míst.[9, strany 104, 206] Touto řadou se inspirovali bratři Chudnovští a v roce 1987 objevili řadu, která produkuje 14 číslic za 1 prvek[14, strana 254]:

$$\frac{1}{\pi} = \frac{12}{640320^{\frac{3}{2}}} \sum_{k=0}^{\infty} \frac{(6k)!(13591409 + 545140134k)}{(3k)!(k!)^3(-640320)^{3k}}.$$

Ta byla několikrát použita k překonání rekordu. Naposledy Timothy Mullicanem v roce 2020, který Ludolphovo číslo vyčíslil na 50 bilionů desetinných číslic.[21] V roce 1995 vymyslel Simon Plouffe algoritmus BBP:

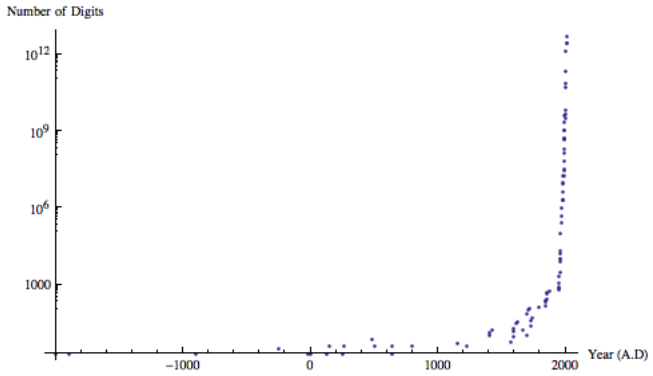
$$\pi = \sum_{k=0}^{\infty} \frac{1}{16^k} \left(\frac{4}{8k+1} - \frac{2}{8k+4} - \frac{1}{8k+5} - \frac{1}{8k+6} \right).$$

[9, strany 117, 126–128] Jedná se o metodu ze skupiny tzv. spigot algoritmů. Tyto algoritmy dokážou spočítat jakoukoli číslici desetinného zápisu čísla Pí, aniž by počítaly ty předchozí.[9, strany 77–84] Algoritmus BBP byl také použit ke zjištění kvadriliontého bitu Ludolphova čísla, což je 0.[9, strana 20] Monte Carlo metody na aproximaci Pí používají pravděpodobnost. Tyto metody ovšem nedávají přesný výsledek, jde pouze o odhady.[9, strana 39] Jednou z těchto metod je Buffonova jehla. Mějme list papíru s nakreslenými rovnoběžnými přímkami. Vzdálenost mezi každými dvěma sousedními přímkami je t. Mějme jehlu délky l a n-krát ji náhodně hodíme na papír. Označme

x počet hodů, při kterých jehla zkřížila některou z přímek. Pak můžeme Pí aproximovat jako:

$$\pi \approx \frac{2nl}{xt}.$$

[22] Při další Monte Carlo metodě máme kruh vepsaný ve čtverci a do čtverce náhodně umisťujeme body. Poměr bodů nacházejících se uvnitř kruhu ku všem umístěným bodům je potom přibližně $\frac{\pi}{4}$[9, strany 39–40].



Obrázek 2.3: Historie přesnosti aproximace Ludolphova čísla [3]

2.4 Využití ve vědě

Číslo Pí se hojně vyskytuje v matematických a fyzikálních rovnicích. Pochopitelně se nachází v geometrii. Např. objem a povrch koule s poloměrem r:

$$V = \frac{4}{3}\pi r^3$$

$$S = 4\pi r^2.$$

V matematice se úhly měří v radiánech. To je jednotka, která je definovaná tak, aby celý kruh měl 2π radiánů. Trigonometrické funkce jako sin a cos mají tím pádem periodu 2π. Každé komplexní číslo z se dá vyjádřit podle jeho absolutní hodnoty r a úhlu ϕ, který znázorňuje rotaci od kladné reálné osy. Tomuto znázornění se říká goniometrický tvar:

$$z = r(\cos\phi + i\sin\phi).$$

Jelikož úhly se vyjadřují v radiánech, často se zde vyskytuje Pí. Za zmínku stojí i s tímto související Eulerova formule, která je mnohými považována za nejkrásnější formuli matematiky:

$$e^{\pi i} + 1 = 0.$$

Ludolphovo číslo se také vyskytuje ve Fourierově transformaci, která slouží k převodu signálu mezi časovou a frekvenční oblastí. Jde o integrální transformaci, která vezme reálnou integrovatelnou funkci f a vrátí funkci:

$$g(\xi) = \int_{-\infty}^{\infty} f(x)e^{2\pi ix\xi}dx.$$

[23, strana 29] Pí se objevuje také u normálního rozdělení, podle kterého se řídí většina jevů v přírodě. Hustotu pravděpodobnosti normálního rozdělení představuje Gaussova funkce a je definována následovně:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}}e^{-\frac{(x-\mu)^2}{2\sigma^2}},$$

kde μ znamená střední hodnotu náhodného rozdělení a σ jeho směrodatnou odchylku.[24, strany 106–107] V mnoha oblastech matematiky se používá Riemannova funkce zeta:

$$\zeta(s) = 1 + \frac{1}{2^s} + \frac{1}{3^s} + \cdots.$$

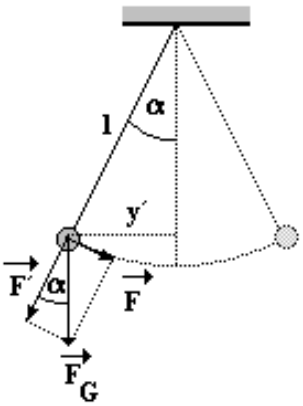
Pro $s = 2$ je výsledek roven $\frac{\pi^2}{6}$.[25, strana 284] Z této skutečnosti vyplývá např. to, že pravděpodobnost, že 2 náhodně zvolená čísla budou nesoudělná, je rovna právě $\frac{\pi^2}{6}$[9, strany 41–43]. Číslo Pí se často používá i ve fyzice. Např. doba kmitu T kyvadla délky l se dá vyjádřit jako:

$$T = 2\pi\sqrt{\frac{l}{g}},$$

kde g je gravitační zrychlení.[26, strana 381] Jedním ze základních pilířů kvantové mechaniky je Heisenbergův princip neurčitosti, který říká, že přesnosti měření polohy částice a její hybnosti jsou na sobě závislé a nemohou být libovolně vysoké. Dá se vyjádřit vztahem:

$$\Delta x \Delta p \geq \frac{h}{4\pi},$$

kde Δx je odchylka měření polohy částice, Δp je odchylka měření hybnosti částice a h je Planckova konstanta.[26, strana 1145]



Obrázek 2.4: Matematické kyvadlo [4]

Kapitola 3

Algoritmy pro výpočet čísla Pí

Budeme se zabývat algoritmy, které se používají pro výpočet čísla Pí. Těchto algoritmů je mnoho. My se zaměříme na ty nejefektivnější a na ty z historického hlediska nejdůležitější. Vybrané algoritmy poté naimplementujeme pomocí knihovny GNU MPFR v jazyce C++. Změříme rychlost jejich konvergence a porovnáme výsledky.

3.1 Algoritmy na principu nekonečných řad

Nejprve se budeme zabývat algoritmy využívající nekonečné řady. Tyto algoritmy se používají k překonání rekordu v počtu vyčíslených cifer Ludolfova čísla nejčastěji. Ačkoli mohou existovat i rychlejší algoritmy, výhodou těchto řad je to, že algoritmy které je využívají, jsou málo náročné na paměť. První nekonečné řady vznikaly v Indii v 15. století. Nejprve byly zaznamenány bez důkazů, ale ty byly zmíněny v díle, které se datuje kolem roku 1530. V Evropě byly tyto řady objeveny až na přelomu 16. a 17. století.[5, strany 101–102]

3.1.1 Wallisův produkt

Nekonečný produkt:

$$\frac{\pi}{2} = (\frac{2}{1} \cdot \frac{2}{3})(\frac{4}{3} \cdot \frac{4}{5})(\frac{6}{5} \cdot \frac{6}{7}) \cdots$$

byl objeven Johnem Wallisem v roce 1656.[9, strana 187] Jedná se o jednu z nejstarších nekonečných řad používaných k aproximaci Pí. Wallisův produkt konverguje velmi pomalu. Na vyčíslení Ludolfova čísla alespoň na jedno desetinné místo je třeba 19 členů produktu. Při objevu se Wallis zabýval následujícím integrálem:

$$\int_0^1 (1 - x^2)^{\frac{1}{2}} dx.$$

Tento integrál je roven obsahu čtvrtiny jednotkového kruhu, tedy $\frac{\pi}{4}$. Wallis tento případ zobecnil na

$$\int_0^1 (1 - x^{\frac{1}{p}})^q dx.$$

Přišel na to, že platí:

$$\int_0^1 (1 - x^{\frac{1}{p}})^q dx = \frac{p!q!}{(p+q)!},$$

pokud p a q jsou celá nezáporná čísla. Dále zjistil, že podobná pravidla platí i pro zlomky. Pokud $p = q = \frac{1}{2}$, jedná se o obsah čtvrtiny jednotkového kruhu, dostáváme tedy:

$$((\frac{1}{2})!)^2 = \frac{\pi}{4}$$

$$(\frac{1}{2})! = \frac{\sqrt{\pi}}{2}.$$

Následně odvodil výsledky pro další racionální hodnoty p a q a díky tomu našel π v nekonečném produktu.[27]

3.1.2 Gregory-Leibnizova řada

Řada:

$$\frac{\pi}{4} = 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} \cdots$$

byla nezávisle na sobě objevena Gottfriedem Leibnizem v roce 1674 a Jamesem Gregorym v roce 1671. Je možné, že v Indii byla známa již v 15. století. Jedná se o speciální případ Gregoryho řady:

$$\arctan z = z - \frac{z^3}{3} + \frac{z^5}{5} - \frac{z^7}{7} \cdots$$

pro z = 1, která byla objevena právě Jamesem Gregorym v roce 1668.[28, strana 247] Gregory-Leibnizova řada vznikla v době, kdy ještě nebyla úplně známa pravidla kalkulu. Její objev proto ovlivnil problémy jako kvadratura kruhu nebo nalezení obsahu plochy pod křivkou. Leibniz se ve skutečnosti zabýval právě problémem kvadratury kruhu, když objevil tuto řadu.[5] Gregory-Leibnizova řada dává při použití 10 členů postupně výsledky (zaokrouhleně): 4; 2,667; 3,466; 2,895; 3,339; 2,976; 3,283; 3,017; 3,252; 3,041. Z toho je trochu vidět, že tato řada konverguje velice pomalu. Při použití 1 000 000 členů řady, dostaneme číslo Pí pouze na 10 desetinných míst. Konvergenci lze značně urychlit použitím jistých technik, kterými se v této práci však nebudeme zabývat. Zvláštností této řady je to, že někdy vyčíslí Ludolfovo číslo na mnohem více desetinných míst s chybami v některých cifrách. Tyto chyby lze předvídat, načemž je založena jedna z technik urychlujících konvergenci.[29,

strany 28–30] Moderní důkaz Gregory-Leibnizovi řady využívá kalkulus. Nejprve musíme dokázat následující rovnici:

$$\frac{1}{1+t^2} = 1 - t^2 + t^4 - t^6 + \cdots + t^{4n} - \frac{t^{4n+2}}{1+t^2}.$$

Podle součtu geometrické řady dostaneme:

$$\frac{1-(-t^2)^{2n+1}}{1-(-t^2)} = \sum_{k=0}^{2n}(-t^2)^k$$

$$\frac{1+(t^2)^{2n+1}}{1+(t^2)} = 1 - t^2 + t^4 - t^6 + \cdots + t^{4n}$$

$$\frac{1}{1+t^2} + \frac{t^{4n+2}}{1+t^2} = 1 - t^2 + t^4 - t^6 + \cdots + t^{4n}$$

$$\frac{1}{1+t^2} = 1 - t^2 + t^4 - t^6 + \cdots + t^{4n} - \frac{t^{4n+2}}{1+t^2}.$$

Odmocnina z reálného čísla je vždy nezáporná, a tedy platí:

$$t^2 \geq 0$$

$$-t^2 \leq 0$$

$$-t^2 \neq 1$$

pro všechna reálná t. Podmínky pro součet geometrické řady jsou splněny a důkaz platí pro všechna reálná t. Nyní mějme reálné číslo $x \in \mathbb{R} : 0 \leq x \leq 1$. Dokázanou rovnici zintegrujeme podle t od 0 do x:

$$\int_0^x \frac{1}{1+t^2}dt = x - \frac{x^3}{3} + \frac{x^5}{5} - \frac{x^7}{7} + \cdots + \frac{x^{4n+1}}{4n-1} - R_n(x)$$

kde

$$R_n(x) = \int_0^x \frac{t^{4n+2}}{1+t^2}dt.$$

Protože odmocnina z reálného čísla je vždy nezáporná máme:

$$t^2 \geq 0$$

$$1 \leq 1 + t^2.$$

Podle pravidel určitého integrálu platí:

$$0 \leq R_n(x) \leq \int_0^x t^{4n+2}dt$$

$$0 \leq R_n(x) \leq \frac{x^{4n+3}}{4n+3}.$$

Přitom $0 \leq x \leq 1$ tedy:

$$\frac{x^{4n+3}}{4n+3} \leq \frac{1}{4n+3}$$

$$0 \leq R_n(x) \leq \frac{1}{4n+3}.$$

Triviálně platí:

$$\lim_{n\to\infty} \frac{1}{4n+3} = 0.$$

Tudíž dostáváme:

$$\int_0^x \frac{1}{1+t^2} dt = x - \frac{x^3}{3} + \frac{x^5}{5} - \frac{x^7}{7} \cdots .$$

Současně víme, že:

$$\frac{d}{dx} \arctan(t) = \frac{1}{1+t^2}$$

$$\arctan(x) = \int_0 x \frac{1}{1+t^2} dt.$$

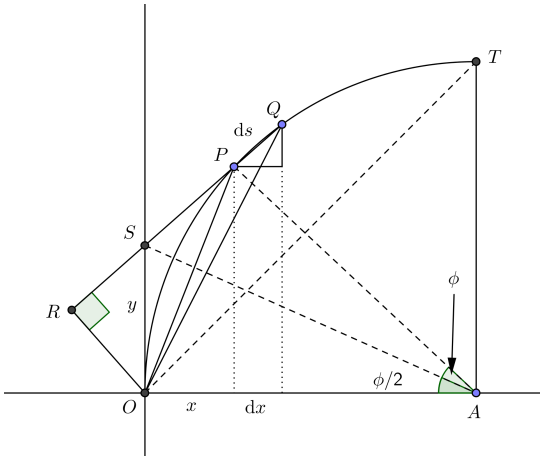
Dostáváme tedy:

$$\arctan(x) = x - \frac{x^3}{3} + \frac{x^5}{5} - \frac{x^7}{7} + \cdots .$$

Pokud za x dosadíme 1:

$$\frac{\pi}{4} = 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} \cdots . \square$$

Nicméně Leibniz řadu dokazoval úplně jinak. Použil k tomu geometrii a goniometrické funkce.[5]



Obrázek 3.1: Základ Leibnizova důkazu [5]

3.1.3 Nilakanthova řada

Další relativně jednoduchou řadou je řada:

$$\pi = 3 + \frac{4}{2 \cdot 3 \cdot 4} - \frac{4}{4 \cdot 5 \cdot 6} + \frac{4}{6 \cdot 7 \cdot 8} + \cdots .$$

Byla obejvena Nilakanthou již v 15. století.[9, strana 223] Nilakantha byl významný indický astronom a matematik z kiralské školy. Nilakanthova řada konverguje mnohem rychleji než Gregory-Leibnizova řada a k vyčíslení čísla Pí je tedy vhodnější. Při použití 10 členů dostaneme výsledek 3,141406 . . . , který je dobře na 3 desetinná místa. Při 100 členech dostáváme strávně 6 desetinných míst a například při použití 100 000 členů dosáhneme na 14 desetinných míst. Zajímavé je, že každý ze jmenovatelů řady představuje obsah pythagorejského trojúhelníku. Všechny pythagorejské trojice mohou být vyjádřeny jako:

$$a = 2mn, b = m^2 - n^2, c = m^2 + n^2,$$

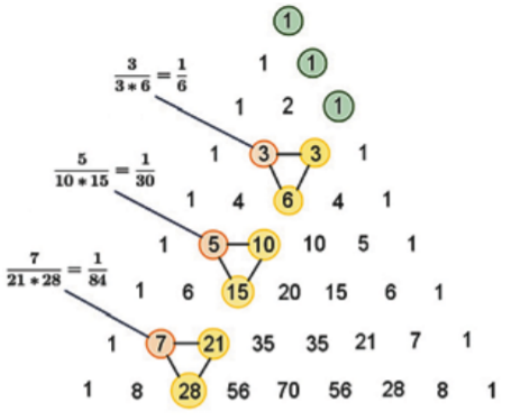
kde m a n jsou celá čísla. Obsah odpovídajícího trojúhelníku je potom

$$S = ab/2 = mn(m^2 - n^2),$$

což pro $n = 1$ dává

$$S = (m - 1)m(m + 1),$$

což přesně odpovídá jmenovatelům Nilakanthovy řady. Tato řada, a díky ní celé Ludolfovo číslo, se také nachází v Pascalově trojůhelníku. Číslo 3 před desetinnou čárkou může být vyjádřeno jako součet tří jedniček (na obrázku 3.2 zeleně) a jednotlivé zlomky se dají vyjádřit jako čísla v oranžovém v čitateli a čísla ve žlutém ve jmenovateli.[6]



Obrázek 3.2: Nilakanthova řada v Pascalově trojúhelníku [6]

3.1.4 Eulerova řada

Eulerovou řadou v kontextu aproximace Ludolfova čísla myslíme řadu:

$$\frac{\pi^2}{6} = 1 + \frac{1}{2^2} + \frac{1}{3^2} + \frac{1}{4^2} + \cdots .$$

Tato řada nekonverguje nijak zvlášť rychle, při 1000 iteracích algoritmu dostaneme pouze 2 desetinné číslice, k výpočtu čísla Pí je tedy nevhodná. Nicméně její objevení mělo vliv na mnoho matematických otázek, například Riemannovy zeta-hypotézy, kterou se matematici zabývají dodnes. Eulerova řada je ve skutečnosti řešením tzv. basilejského problému. Jedná se právě o součet řady:

$$1 + \frac{1}{2^2} + \frac{1}{3^2} + \frac{1}{4^2} + \cdots .$$

Tento problém byl poprvé vysloven v roce 1650 a vyřešen právě Leonardem Eulerem v roce 1735. Podle Eulerovy práce byla v roce 1859 definována Riemannova zeta-funkce. Operace, které Euler při objevu prováděl, nebyly v té době korektně dokázány, ale o 100 let později Karl Weierstrass prokázal, že Eulerovy úpravy byly korektní. I bez tohoto potvrzení byl Euler schopen své řešení dokázat numericky pomocí částečných součtů řady a v roce 1741 vydal podrobný důkaz. Řešení využívá Taylorovu řadu pro funkci sinus:

$$\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \cdots .$$

[30]

3.1.5 Machin-like formule

Machinova formule:

$$\frac{\pi}{4} = 4 \arctan \frac{1}{5} - \arctan \frac{1}{239}$$

byla objevena v roce 1706. Machin využil Gregoryho řadu a vytvořil formuli, která konverguje mnohem rychleji než doposud známé řady. Již po 4 krocích algoritmu dostáváme číslo Pí na 5 desetinných míst. Po 10 krocích jsme již na 14 desetinných místech. Machin tuto formuli použil pro vyčíslení Ludolfova čísla na 100 desetinných míst.[31, strana 102] V průběhu dalších let vznikaly nové formule tvaru:

$$c_0 \frac{\pi}{4} = \sum_{n=1}^{N} c_n \arctan \frac{a_n}{b_n},$$

tzv. Machin-like formule. Tyto formule hrály významnou roli ve vyčíslování Pí až do 2. poloviny 20. století. Machin-like formule využívají Taylorovu řadu pro funkci arctan [32]:

$$\arctan x = x - \frac{x^3}{3} + \frac{x^5}{5} - \frac{x^7}{7} + \cdots .$$

Dalšími významnými Machin-like formulemi jsou Shanksova formule:

$$\frac{\pi}{4} = 6\arctan\frac{1}{8} + 2\arctan\frac{1}{57} + \arctan\frac{1}{239},$$

pomocí níž bylo v roce 1961 vypočteno 100 000 číslic π[33] a Takanova formule, díky níž bylo Ludolfovo číslo vyčísleno na více než bilion desetinných míst:

$$\frac{\pi}{4} = 12\arctan\frac{1}{49} + 32\arctan\frac{1}{57} - 5\arctan\frac{1}{239} + 12\arctan\frac{1}{110443}.$$

Pro důkaz Machinovy formule musíme nejdřive dokázat tangens součtu a tangens rozdílu:

$$\tan(u-v) = \frac{\tan u - \tan v}{1 + \tan u \tan v}$$

pro všechna u a v. Víme, že: $\sin(u-v) = \sin u\cos v - \sin v\cos u$ a $\cos(u-v) = \cos u\cos v + \sin u\sin v$. Tangens rozdílu můžeme tedy přepsat jako:

$$\tan(u-v) = \frac{\sin(u-v)}{\cos(u-v)} = \frac{\sin u\cos v - \sin v\cos u}{\cos u\cos v + \sin u\sin v}.$$

Vydělením čitatele i jmenovatele $\cos u\cos v$ za předpokladu $u, v \neq \frac{\pi}{2} + \pi n$, pro všechna $n \in \mathbb{N}$, dostaneme:

$$\tan(u-v) = \frac{\frac{\sin u\cos v - \sin v\cos u}{\cos u\cos v}}{\frac{\cos u\cos v + \sin u\sin v}{\cos u\cos v}} = \frac{\tan u - \tan v}{1 + \tan u\tan v}.$$

Tím je tangens rozdílu dokázán. Pro tangens součtu:

$$\tan(u+v) = \frac{\tan u + \tan v}{1 - \tan u\tan v}$$

stačí místo v dosadit $-v$ a využít faktu, že tangens je lichá funkce. Nyní můžeme dokázat Machinovu formuli:

$$\frac{\pi}{4} = 4\arctan\frac{1}{5} - \arctan\frac{1}{239}.$$

Aplikujeme funkci tangens na obě strany rovnice:

$$\tan\frac{\pi}{4} = \tan(4\arctan\frac{1}{5} - \arctan\frac{1}{239}).$$

Levá strana je rovna 1 a pravou stranu můžeme pomocí tangensu rozdílu přepsat jako:

$$\frac{\tan(4\arctan\frac{1}{5}) - \tan(\arctan\frac{1}{239})}{1 + \tan(4\arctan\frac{1}{5})\tan(\arctan\frac{1}{239})} = \frac{\tan(4\arctan\frac{1}{5}) - \frac{1}{239}}{1 + \tan(4\arctan\frac{1}{5})\frac{1}{239}}.$$

Dále můžeme $\tan(4\arctan\frac{1}{5})$ zapsat jako:

$$\tan(4\arctan\frac{1}{5}) = \tan(2\arctan\frac{1}{5} + 2\arctan\frac{1}{5})$$

a díky tangensu součtu dostaneme:

$$\tan(4\arctan\frac{1}{5}) = \frac{\tan(2\arctan\frac{1}{5}) + \tan(2\arctan\frac{1}{5})}{1 - \tan(2\arctan\frac{1}{5})\tan(2\arctan\frac{1}{5})}$$

$$\tan(4\arctan\frac{1}{5}) = \frac{\tan(\arctan\frac{1}{5} + \arctan\frac{1}{5}) + \tan(\arctan\frac{1}{5} + \arctan\frac{1}{5})}{1 - \tan(\arctan\frac{1}{5} + \arctan\frac{1}{5})\tan(\arctan\frac{1}{5} + \arctan\frac{1}{5})}.$$

Po opětovném použití tangensu součtu máme:

$$\tan(4\arctan\frac{1}{5}) = \frac{\frac{\tan(\arctan\frac{1}{5})+\tan(\arctan\frac{1}{5})}{1-\tan(\arctan\frac{1}{5})\tan(\arctan\frac{1}{5})} + \frac{\tan(\arctan\frac{1}{5})+\tan(\arctan\frac{1}{5})}{1-\tan(\arctan\frac{1}{5})\tan(\arctan\frac{1}{5})}}{1 - \frac{\tan(\arctan\frac{1}{5})+\tan(\arctan\frac{1}{5})}{1-\tan(\arctan\frac{1}{5})\tan(\arctan\frac{1}{5})} \cdot \frac{\tan(\arctan\frac{1}{5})+\tan(\arctan\frac{1}{5})}{1-\tan(\arctan\frac{1}{5})\tan(\arctan\frac{1}{5})}}$$

a protože $\tan(\arctan\frac{1}{5}) = \frac{1}{5}$ dostaneme:

$$\tan(4\arctan\frac{1}{5}) = \frac{\frac{\frac{1}{5}+\frac{1}{5}}{1-\frac{1}{5}\cdot\frac{1}{5}} + \frac{\frac{1}{5}+\frac{1}{5}}{1-\frac{1}{5}\cdot\frac{1}{5}}}{1 - \frac{\frac{1}{5}+\frac{1}{5}}{1-\frac{1}{5}\cdot\frac{1}{5}} \cdot \frac{\frac{1}{5}+\frac{1}{5}}{1-\frac{1}{5}\cdot\frac{1}{5}}} = \frac{2\frac{50}{120}}{1-(\frac{50}{120})^2} = \frac{\frac{5}{6}}{1-\frac{2500}{14400}} = \frac{72000}{71400} = \frac{120}{119}.$$

Po dosazení získáme:

$$\frac{\frac{120}{119} - \frac{1}{239}}{1 + \frac{120}{119}\cdot\frac{1}{239}} = \frac{\frac{28561}{28441}}{\frac{28561}{28441}} = 1,$$

což znamená že:

$$\frac{\pi}{4} = 4\arctan\frac{1}{5} - \arctan\frac{1}{239}.\square$$

[32]

3.1.6 Chudnovskyho algoritmus

Algoritmus bratří Chudnovských je založen na formuli:

$$\frac{1}{\pi} = \frac{12}{640320^{\frac{3}{2}}}\sum_{k=0}^{\infty}\frac{(6k)!(13591409+545140134k)}{(3k)!(k!)^3(-640320)^{3k}}.$$

[34] Tato formule vychází z práce Ramanujana, který vytvořil několik formulí pro výpočet π tvaru:

$$\frac{1}{\pi} = \sum_{k=0}^{\infty} s(k)\frac{Ak+B}{C^k}$$

kde *s(k)* je posloupnost celých čísel a A, B a C jsou konstanty. Těmto fromulím se říká Ramanujan-Sato řady.[35] Bratři Chudnovští se jako mnoho matematiků před nimi zabývali otázkami jako je Ludolfovo číslo normální, jak vypadá jeho reprezentace pomocí nekonečného řetězového zlomku a jak moc je číslo Pí iracionální.[36] Svoji formuli, která produkuje více než 14 desetinných míst Ludolfova čísla za jeden člen, publikovali v roce 1988. V současné době se jedná

o nejrychlejší algoritmus pro aproximaci π. Byl několikrát použit k překonání světového rekordu: 2,7 bilionu desetinných číslic Pí v roce 2009[37], 10 bilionů v roce 2011[38], 22,4 bilionů číslic v roce 2016[39], 31,4 bilionů v letech 2018–2019[40] a 50 bilionů číslic v lednu 2020[21]. Sami bratři Chudnovští použili svůj algoritmus v letech 1988–1989 k aproximaci π na 480 milionů desetinných míst. Jejich kódy byly napsány v jazyce Fortran a počítání probíhalo na počítačích IBM.[36] Formule může být zjednodušena:

$$\frac{1}{\pi} = \frac{12}{640320^{\frac{3}{2}}} \sum_{k=0}^{\infty} \frac{(6k)!(13591409 + 545140134k)}{(3k)!(k!)^3(-640320)^{3k}}$$

$$\frac{1}{\pi} = \frac{12}{640320\sqrt{640320}} \sum_{k=0}^{\infty} \frac{(6k)!(13591409 + 545140134k)}{(3k)!(k!)^3(-640320)^{3k}}$$

$$\frac{1}{\pi} = \frac{1}{426880\sqrt{10005}} \sum_{k=0}^{\infty} \frac{(6k)!(13591409 + 545140134k)}{(3k)!(k!)^3(-640320)^{3k}}$$

$$\frac{1}{\pi} = \frac{1}{426880\sqrt{10005}} \sum_{k=0}^{\infty} \frac{(6k)!(13591409 + 545140134k)}{(3k)!(k!)^3(-262537412640768000)^{k}}.$$

V tomto vzorci můžeme vidět 3 členy: multinomiální člen M_k, lineární člen L_k, exponenciální člen X_k a konstantu C pro něž platí:

$$M_k = \frac{(6k)!}{(3k)!(k!)^3}$$

$$L_k = 545140134k + 13591409$$

$$X_k = (-262537412640768000)^k.$$

$$C = 426880\sqrt{10005}.$$

Pro multinomiální člen dále platí:

$$M_{k+1} = \frac{(6k+6)!}{(3k+3)!((k+1)!)^3}$$

$$M_{k+1} = \frac{(6k+6)(6k+5)\cdots(6k+1)(6k)!}{(3k+3)(3k+2)(3k+1)(3k)!(k+1)^3(k!)^3}$$

$$M_{k+1} = \frac{(6k)!}{(3k)!(k!)^3} \cdot \frac{(6k+6)(12k+10)(6k+4)(12k+6)(6k+2)(12k+2)}{(6k+6)(6k+4)(6k+2)(k+1)^3}$$

$$M_{k+1} = \frac{(6k)!}{(3k)!(k!)^3} \cdot \frac{(12k+10)(12k+6)(12k+2)}{(k+1)^3}$$

Nyní už můžeme určit rekurentní vztahy:

$$M_{k+1} = M_k \frac{(12k+10)(12k+6)(12k+2)}{(k+1)^3}$$

$$L_{k+1} = L_k + 545140134$$

$$X_{k+1} = X_k(-262537412640768000)$$

s počátečními podmínkami:

$$M_0 = 1, L_0 = 13591409, X_0 = 1.$$

Číslo Pí vyjádříme jako:

$$\pi = C(\sum_{k=0}^{\infty} \frac{M_k L_k}{X_k})^{-1}.$$

3.2 Ostatní algoritmy

V této části se budeme zabývat různými algoritmy k aproximaci π, které nepoužívají nekonečné řady. Mezi ně patří iterativní algoritmy. Zatímco algoritmy na principu nekonečných řad s každým členem přidají několik desetinných míst, iterativní algoritmy jejich počet znásobí. Konvergují tedy značně rychleji než nekonečné řady. Jejich nevýhodou ovšem je, že jsou značně náročné na paměť. První takové algoritmy vznikaly v letech 1975–1976. Dalšími algoritmy jsou tzv. spigot algoritmy. Pomocí nich lze zjistit určitou číslici Ludolfova čísla bez nutnosti znát všechny předchozí. Tyto metody se používají k ověření správnosti nových rekordů. Když vznikne nová aproximace, pomocí některého spigot algoritmu se určí několik náhodných číslic z konce aproximace. Pokud se shodují, rekord je ověřen. Spigot algoritmy vznikaly v 90. letech 20. století.

3.2.1 Gauss–Legendreho algoritmus

Tento algoritmus je založen na práci Gausse a Legendreho z 19. století. Jeho moderní verze byla objevena v roce 1975 Richardem Brentem a Eugenem Salaminem. Proto je někdy také nazýván Brent–Salamin algoritmus.[41] Algoritmus je založen na následujících rekurentních vztazích:

$$a_{n+1} = \frac{a_n + b_n}{2}$$

$$b_{n+1} = \sqrt{a_n b_n}$$

$$t_{n+1} = t_n - p_n(a_n - a_{n+1})^2$$

$$p_{n+1} = 2p_n$$

s počátečními podmínkami:

$$a_0 = 1$$

$$b_0 = \frac{1}{\sqrt{2}}$$

$$t_0 = \frac{1}{4}$$
$$p_0 = 1.$$

Potom je možné aproximovat Pí jako:

$$\pi \approx \frac{(a_n + b_n)^2}{4t_n}.$$

[42] Algoritmus s každým krokem počet správných číslic desetinného zápisu π zhruba zdvojnásobí. Již při třetí iteraci dostáváme aproximaci správnou na 18 desetinných míst, algoritmus tedy konverguje velice rychle. Hlavní myšlenkou algoritmu je Gaussem definovaný aritmeticko-geometrický průměr dvou čísel. Mějme čísla a a b a definujme posloupnosti a_n a b_n jako:

$$a_0 = a$$
$$a_{n+1} = \frac{a_n + b_n}{2}$$

a

$$b_0 = b$$
$$b_{n+1} = \sqrt{a_n b_n}.$$

Potom můžeme aritmeticko-geometrický průměr definovat jako:

$$M(a, b) = \lim_{n\to\infty} a_n = \lim_{n\to\infty} b_n.$$

[41]

3.2.2 Borwein algoritmy

Dalším iterativním algoritmem je algoritmus bratří Borweinů. Jonathan a Peter Borweinovi se zabývali aproximací Ludolfova čísla mnoho let. Inspirovali se prací Ramanujana a vymysleli mnoho algoritmů.[43] První z nich vytvořili v roce 1984. Mějme počáteční podmínky:

$$a_0 = \sqrt{2}$$
$$b_0 = 0$$
$$p_0 = 2 + \sqrt{2},$$

potom můžeme algoritmus popsat rekurencemi:

$$a_{n+1} = \frac{\sqrt{a_n} + \frac{1}{\sqrt{a_n}}}{2}$$
$$b_{n+1} = \frac{(1 + b_n)\sqrt{a_n}}{a_n + b_n}$$

$$p_{n+1} = \frac{(1 + a_{n+1})p_n b_{n+1}}{1 + b_{n+1}}.$$

Posloupnost p_k potom konverguje s řádem konvergence 2 ($r = 2$) k π. S každou iterací se tedy počet správných číslic přibližně zdvojnásobí.[9, strana 236] V průběhu dalších let Borweinovi přicházeli na další algoritmy. Algoritmus z roku 1985 s opět kvadratickou konvergencí($r = 2$) byl v roce 1986 použit ke spočtení 29 milionů číslic Pí[44] a v roce 2009 k vyčíslení π na 2.5 bilionu desetinných míst.[45] V roce 1987 bratři objevili další kvadratický algoritmus. V roce 1991 přišli na kubický($r = 3$) algoritmus, který počet správných číslic při každém kroku ztrojnásobí:

$$r_{k+1} = \frac{3}{1 + 2(1 - s_k^3)^{\frac{1}{3}}}$$

$$s_{k+1} = \frac{r_{k+1} - 1}{2}$$

$$a_{k+1} = r_{k+1}^2 a_k - 3^k(r_{k+1}^2 - 1),$$

Přičemž

$$a_0 = \frac{1}{3}$$

$$s_0 = \frac{\sqrt{3} - 1}{2}.$$

Potom platí:

$$\pi \approx \frac{1}{a_k}.$$

Za zmínku stojí také quartický algoritmus:

$$y_{k+1} = \frac{1 - (1 - y_k^4)^{\frac{1}{4}}}{1 + (1 - y_k^4)^{\frac{1}{4}}}$$

$$a_{k+1} = a_k(1 + y_{k+1})^4 - 2^{2k+3} y_{k+1}(1 + y_{k+1} + y_{k+1}^2),$$

kde

$$a_0 = 2(\sqrt{2} - 1)^2$$

$$y_0 = \sqrt{2} - 1.$$

a_k pak konverguje s řádem konvergence 4 ($r = 4$) k hodnotě $\frac{1}{\pi}$, počet správných cifer se tedy s každou iterací zečtyřnásobí. Jeden krok quartického algoritmu je ekvivalentní dvěma krokům Gauss–Legendreho algoritmu.

Borweinovi vymysleli také quintický($r = 5$) a nonický($r = 9$) algoritmus, který počet číslic v každé iteraci zpětinásobí, respektive zdevítinásobí. Ačkoli tyto algoritmy konvergují rychleji z pohledu počtu potřebných iterací, díky složitým výpočtům jsou pomalejší z pohledu počtu správně aproximovaných číslic za určitý čas.[42]

3.2.3 algoritmus BBP

Formule:

$$\pi = \sum_{i=0}^{\infty} \frac{1}{16^i}\left(\frac{4}{8i+1} - \frac{2}{8i+4} - \frac{1}{8i+5} - \frac{1}{8i+6}\right)$$

byla objevena v roce 1995 Plouffem a publikována o rok později Baileym, Borweinem a Plouffem.[46] Tato formule patří do rodiny spigot algoritmů. Lze pomocí ní zjistit jakoukoli hexadecimální cifru Pí, aniž by bylo nutné počítat všechny předchozí cifry.

Objevitelé BBP formule se zabývali formulí:

$$\log 2 = \sum_{k=0}^{\infty} \frac{1}{k2^k},$$

díky které lze zjistit jakoukoli binární číslici $\log 2$. Snažili se najít podobné formule i pro jiné konstanty. Toto hledání prováděli pomocí Fergusonova algoritmu PSLQ na hledání spojitostí mezi celými čísly. Tento algoritmus je považován za jeden z největších objevů 20. století v oblasti experimentální matematiky. Hledání pomocí PSLQ bylo úspěšné, a tak byla nalezena BBP formule.[47]

Později byly objevovány podobné formule tvaru:

$$\alpha = \sum_{k=0}^{\infty}\left(\frac{1}{b^k} \cdot \frac{p(k)}{q(k)}\right)$$

pro další konstanty α, kde $p(k)$ a $q(k)$ jsou polynomy s celočíselnými koeficienty a b je celočíselná báze. K hledání polynomů $p(k)$ a $q(k)$ se používá právě PSLQ algoritmus.[48]

Specializací obecné formule je:

$$P(s, b, m, A) = \sum_{k=0}^{\infty}\left(\frac{1}{b^k}\sum_{j=1}^{m}\frac{a_j}{(mk+j)^s}\right),$$

kde s, b a m jsou celá čísla a $A = (a_1, a_2, \ldots, a_m)$ je celočíselná posloupnost. Pro BBP formuli platí:

$$P(1, 16, 8, (4, 0, 0, -2, -1, -1, 0, 0)) = \sum_{i=0}^{\infty} \frac{1}{16^i}\left(\frac{4}{8i+1} - \frac{2}{8i+4} - \frac{1}{8i+5} - \frac{1}{8i+6}\right).$$

Existuje také formule tohoto tvaru pro π^2 [44]:

$$\pi^2 = \sum_{i=0}^{\infty} \frac{1}{16^i}\left(\frac{16}{(8i+1)^2} - \frac{16}{(8i+2)^2} - \frac{8}{(8i+3)^2} - \frac{16}{(8i+4)^2}\right.$$
$$\left. - \frac{4}{(8i+5)^2} - \frac{4}{(8i+6)^2} + \frac{2}{(8i+7)^2}\right).$$

V roce 1997 Fabrice Bellard BBP formuli mírně modifikoval a dosáhl tak asi 43% zrychlení:

$$\pi = -\frac{1}{64}\sum_{k=0}^{\infty}\frac{(-1)^k}{1024^k}(\frac{32}{4k+1}+\frac{8}{4k+2}+\frac{1}{4k+3}).$$

[47]

Pro důkaz BBP formule začneme následující rovnicí, která platí pro všechna $k < 8$:

$$\int_0^{\frac{1}{\sqrt{2}}}\frac{x^{k-1}}{1-x^8}dx = \int_0^{\frac{1}{\sqrt{2}}}x^{k-1}(\sum_{i=0}^{\infty}(x^8)^i)dx = \sum_{i=0}^{\infty}(\int_0^{\frac{1}{\sqrt{2}}}x^{k-1+8i}dx).$$

Úpravu podle součtu geometrické řady můžeme provést, protože $|x^8| < 1$ pro každé x, $0 \leq x \leq \frac{1}{\sqrt{2}}$. Po úpravě dostaneme:

$$\int_0^{\frac{1}{\sqrt{2}}}\frac{x^{k-1}}{1-x^8}dx = \sum_{i=0}^{\infty}[\frac{x^{k+8i}}{k+8i}]_{x=0}^{x=\frac{1}{\sqrt{2}}} = \frac{1}{2^{\frac{k}{2}}}\sum_{i=0}^{\infty}\frac{1}{16^i(8i+k)}.$$

Díky tomu můžeme BBP formuli zapsat jako:

$$\sum_{i=0}^{\infty}\frac{1}{16^i}(\frac{4}{8i+1}-\frac{2}{8i+4}-\frac{1}{8i+5}-\frac{1}{8i+6}) = \int_0^{\frac{1}{\sqrt{2}}}\frac{4\sqrt{2}-8x^3-4\sqrt{2}x^4-8x^5}{1-x^8}dx.$$

Nyní provedeme substituci $y = \sqrt{2}x$, tedy $x = \frac{1}{\sqrt{2}}$ a $dx = \frac{1}{\sqrt{2}}dy$. Po úpravě máme:

$$\sum_{i=0}^{\infty}\frac{1}{16^i}(\frac{4}{8i+1}-\frac{2}{8i+4}-\frac{1}{8i+5}-\frac{1}{8i+6})$$

$$= \int_0^1\frac{4\sqrt{2}-8\cdot\frac{1}{2\sqrt{2}}y^3-4\sqrt{2}\cdot\frac{1}{4}y^4-8\cdot\frac{1}{4\sqrt{2}}y^5}{1-\frac{1}{16}y^8}\cdot\frac{1}{\sqrt{2}}dy$$

$$= 16\int_0^1\frac{y^5+y^4-2y^3-4}{y^8-16}dy = 16\int_0^1\frac{y-1}{(y^2-2)(y^2-2y+2)}dy.$$

Po použití metody parciálních zlomků dostaneme:

$$\sum_{i=0}^{\infty}\frac{1}{16^i}(\frac{4}{8i+1}-\frac{2}{8i+4}-\frac{1}{8i+5}-\frac{1}{8i+6})$$

$$= 2\int_0^1\frac{2y}{y^2-2}dy - 2\int_0^1\frac{2y-2}{y^2-2y+2}dy + 4\int_0^1\frac{1}{y^2-2y+2}dy$$

$$= -4\arctan(-1) = \pi.\square$$

[49]

3.2.4 Rabinowitz–Wagon algoritmus

První spigot algoritmus vymysleli v roce 1990 Rabinowitz a Wagon. Vycházeli při tom z této řady:

$$\pi = \sum_{i=0}^{\infty} \frac{(i!)^2 2^{i+1}}{(2i+1)!},$$

kterou je možné odvodit z Wallisova produktu.

Algoritmus začíná s dvojkami ve sloupcích označených zlomky, jak je znázorněno na obrázku 3.3. Každá dvojka je vynásobena deseti. Potom je zprava každý výsledek zredukován podle modula *den*, přičemž $\frac{num}{den}$ je označení daného sloupce. Vzniká kvocient q a zbytek r. Zbytek je ponechán do další iterace a výsledek výrazu $q \cdot num$ je přičten k výsledku dalšího sloupce. Potom číslice na místě desítek výsledku úplně vlevo představuje další číslici π. Algoritmus pokračuje vynásobením všech zbytků deseti a opětovným redukováním.[7]

	Digits of π	1/3	2/5	3/7	4/9	5/11	6/13	7/15	8/17	9/19	10/21	11/23	12/25
Initialize		2	2	2	2	2	2	2	2	2	2	2	2
×10		20	20	20	20	20	20	20	20	20	20	20	20
Carry	3	+10	+12	+12	+12	+10	+12	+7	+8	+9	+0	+0	+0
		30	32	32	32	30	32	27	28	29	20	20	20
Remainders		0	2	2	4	3	10	1	13	12	1	20	20
×10		0	20	20	40	30	100	10	130	120	10	200	200
Carry	1	+13	+20	+33	+40	+65	+48	+98	+88	+72	+150	+132	+96
		13	40	53	80	95	148	108	218	192	160	332	296
Remainders		3	1	3	3	5	5	4	8	5	8	17	20
×10		30	10	30	30	50	50	40	80	50	80	170	200
Carry	4	+11	+24	+30	+40	+40	+42	+63	+64	+90	+120	+88	+0
		41	34	60	70	90	92	103	144	140	200	258	200
Remainders		1	1	0	0	0	4	12	9	4	10	6	16
×10		10	10	0	0	0	40	120	90	40	100	60	160
Carry	1	+4	+2	+9	+24	+55	+84	+63	+48	+72	+60	+66	+0
		14	12	9	24	55	124	183	138	112	160	126	160

Obrázek 3.3: Znázornění 1. spigot algoritmu [7]

Kapitola 4

Implementace algoritmů a porovnání ryhlosti konvergence

V této kapitole se budeme zabývat knihovnou GNU MPFR pro práci s čísly s plovoucí čárkou s neomezenou přesností. Ta byla použita při implementaci algoritmů z předchozí kapitoly. Dále se podrobně podíváme na zdrojové kódy některých ze zmíněných algoritmů. V závěru kapitoly změříme rychlost konvergence algoritmů z pohledu času potřebného pro dosažení určité přesnosti. Výsledky porovnáme a znázorníme pomocí grafů.

4.1 Knihovna GNU MPFR

GNU MPFR je knihovna pro jazyk C a C++ pro aritmetiku s plovoucí čárkou s libovolnou přesností. Je založena na knihovně GNU MP a rozšiřuje IEEE 754 standart, díku tomu jsou výsledky programů nezávislé na platformě. GNU MPFR je zdarma distribuována pod GNU LGPL. Tato knihovna nabízí dobře definovanou sémantiku a správné zaokrouhlování pro všechny implementované matematické funkce. Každé číslo má vlastní přesnost v bitech, která se nastaví při inicializaci proměnné. GNU MPFR podporuje speciální čísla jako nekonečna a NaN (not a number). Funkce implementované v této knihovně využívají efektivní algoritmy, díky kterým dosahuje velké rychlosti. Na obrázku 4.1 můžeme vidět srovnání rychlosti několika základních matematických operací s ostatními knihovnami.[8]

operation	digits	MPFR 2.2.0	CLN 1.1.11	PARI 2.2.12-beta	NTL 5.4
$x \times y$	10^2	**0.00048**	0.00071	0.00056	0.00079
	10^4	**0.48**	0.81	0.58	0.57
x/y	10^2	**0.0010**	0.0013	0.0011	0.0020
	10^4	**1.2**	2.4	**1.2**	**1.2**
$\sqrt{x}$	10^2	**0.0014**	0.0016	0.0015	0.0037
	10^4	**0.81**	1.58	0.82	1.23
$\exp x$	10^2	**0.017**	0.060	0.032	0.140
	10^4	**54**	70	68	1740

Obrázek 4.1: Porovnání rychlosti různých knihoven [8]

Knihovna GNU MPFR dává pro testované funkce nejlepší vysledky, to byl také hlavní důvod, proč byla pro tuto práci vybrána.

4.2 Implementace

Algoritmy byly implementovány pomocí GNU MPFR. Byly použity zejména funkce `mpfr_add`, `mpfr_sub`, `mpfr_mul`, `mpfr_div`, `mpfr_pow` a `mpfr_sqrt`. Postupně se podíváme na implementaci algoritmů využívajících nekonečné řady, iterativních algoritmů a spigot algoritmů. Některé z nich rozebereme podrobněji.

4.2.1 Implementace algoritmů na principu nekonečných řad

Implementace těchto algoritmů funguje tak, že nastaví aproximaci na nějakou počáteční hodnotu. Potom při každém průchodu for cyklem spočítá další člen řady a přičtením nebo vynásobením ho přidá k aproximaci. Na závěr je většinou třeba aproximaci vynásobit nějakým číslem. Důkladně probereme algoritmus využívající Gregory–Leibnizovu řadu, popsaný v kapitole 3.1.2:

$$\frac{\pi}{4} = 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} \cdots .$$

Zdrojový kód je k nalezení v příloze v části A. Implementace nejprve nastaví aproximaci na počáteční hodnotu 0. Potom ve for cyklu vždy nastaví právě počítaný člen řady na 1, vydělí ho odpovídajícím jmenovatelem a vynásobí znaménkem, tedy číslem 1 nebo -1, podle toho, jestli se má přičíst či odečíst. Nakonec se spočtený člen přičte k aproximaci a nastaví se znaménko a jmenovatel na správné hodnoty pro následující iteraci. Na úplný závěr se aproximace vynásobí 4.

4.2.2 Implementace iterativních algoritmů

Iterativní algoritmy jsou založeny na rekurentních vztazích a počátečních podmínkách. Při implementaci je tedy nejdříve nutné nastavit proměnné na počáteční hodnoty podle počátečních podmínek. Dále se při každém průchodu

for cyklem proměnné upraví podle rekurentních vztahů. Aproximace π se pak získá z výše zmiňovaných proměnných. Podrobně se podíváme na implementaci Gauss–Legendreho algoritmu, probraného v části 3.2.1:

$$a_{n+1} = \frac{a_n + b_n}{2}$$

$$b_{n+1} = \sqrt{a_n b_n}$$

$$t_{n+1} = t_n - p_n(a_n - a_{n+1})^2$$

$$p_{n+1} = 2p_n$$

$$a_0 = 1, b_0 = \frac{1}{\sqrt{2}}, t_0 = \frac{1}{4}, p_0 = 1.$$

Zdrojový kód je v části B. Nejdříve se nastaví proměnné `a`, `b`, `t` a `p` podle počátečních podmínek. Dále se ve for cyklu vždy nastaví proměnné `aOld` a `tOld` na hodnoty proměnných `a` a `t` z předchozí iterace a hodnoty proměnných `a`, `b`, `t` a `p` se upraví podle rekurentních vztahů. Výsledný odhad Pí se pak získá podle vztahu $\pi \approx \frac{(a_n+b_n)^2}{4t_n}$.

4.2.3 Implementace spigot algoritmů

Spigot algoritmy v každém průchodu for cyklem přidají k aproximaci jednu číslici. V této práci se zabýváme pouze dvěma algoritmy z této skupiny. Implementace algoritmu BBP je podobná implementaci algoritmů na principu nekonečných řad. Podíváme se tedy na Rabinowitz–Wagon algoritmus, který byl popsán v kapitole 3.2.4.

Zdrojový kód lze nalézt v části C. Implementace nastaví aproximaci na počáteční hodnotu 0 a délku vektorů na potřebnou hodnotu podle zvoleného počtu iterací. Dále se nastaví počáteční hodnoty a pojmenování sloupců. Při každém průchodu for cyklem se potom výsledek vynásobí 10 a přičte se k němu odpovídající carry, uloží se zbytek po zredukování modulem *den* a výraz $q{\cdot}num$ se uloží jako carry pro další sloupec. Výsledek posledního sloupce modulo 10 pak představuje další desetinnou číslici Pí. Získaná číslice je potom vydělena odpovídajícím násobkem 10 a přičtena k aproximaci.

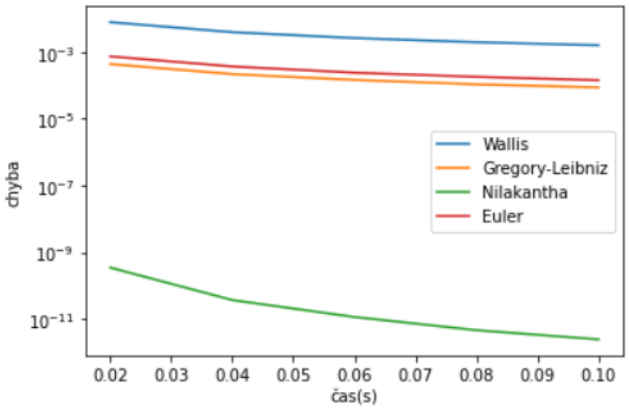
4.3 Porovnání výsledků

Každý algoritmus byl několikrát spuštěn s různým počtem iterací. Vždy byl změřen čas výpočtu a přesnost aproximace. K měření přesnosti aproximace byla použita funkce z knihovny GNU MPFR `mpfr_const_pi`. Algoritmy byly pro porovnání výsledků rozděleny do tří kategorií. Jednoduché nekonečné řady, mezi které byly zařazeny Wallisův produkt, Gregory–Leibnizova řada, Nilakanthova řada a Eulerova řada. Komplexnější nekonečné řady a spigot algoritmy, do kterých byly zařazeny Machin-like řady, Chudnovskyho algoritmus,

algoritmus BBP a Rabinowitz–Wagon algoritmus. A iterativní algoritmy, do kterých patří Gauss–Legendreho algoritmus, Borwein-Quadratic algoritmus, Borwein-Cubic algoritmus a Borwein-Quartic algoritmus.

4.3.1 Porovnání výsledků jednoduchých nekonečných řad

Na obrázku 4.2 je vidět chyba aproximace dosažené za daný čas v sekundách.

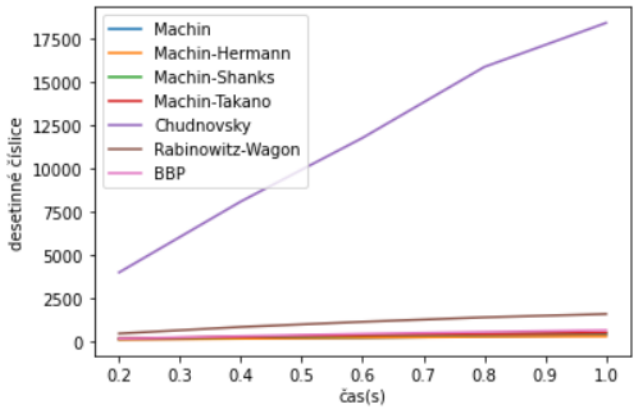


Obrázek 4.2: Porovnání výsledků jednoduchých nekonečných řad

Podle očekávání dopadla nejlépe Nilakanthova řada, popsaná v kapitole 3.1.3, a to výrazně. Ostatní řady jsou na tom dost podobně, Wallisův produkt, popsaný v kapitole 3.1.1 mírně zaostává.

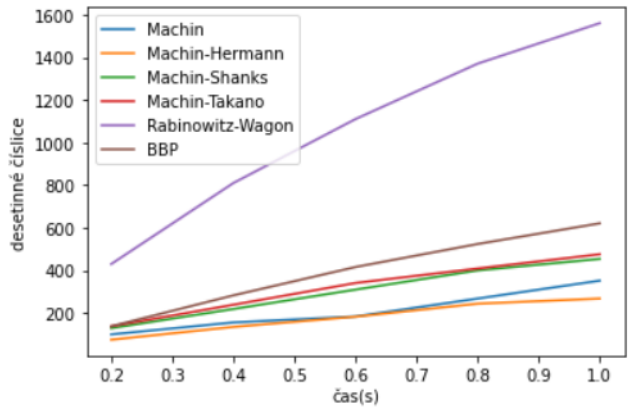
4.3.2 Porovnání výsledků komplexnějších nekonečných řad a spigot algoritmů

Na obrázku 4.3 je znázorněn počet správných desetinných číslic aproximace dosažené za určitý čas v sekundách.



Obrázek 4.3: Porovnání výsledků komplexnějších nekonečných řad a spigot algoritmů

Výrazně nejlepších výsledků dosahuje Chudnovskyho algoritmus, probraný v kapitole 3.1.6. Pro přehlednost výsledků dalších algoritmů se podíváme na obrázek 4.4, do kterého Chudnovskyho algoritmus nebyl zařazen.

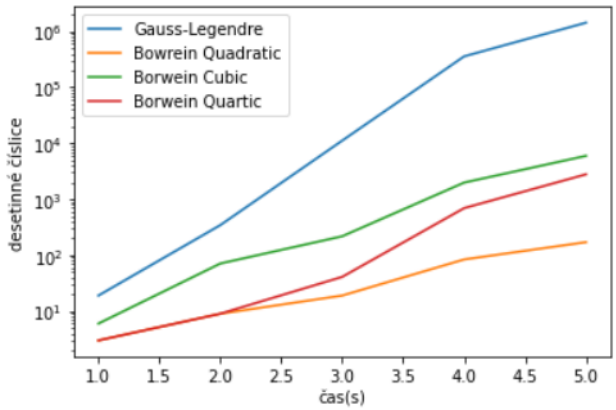


Obrázek 4.4: Porovnání výsledků komplexnějších nekonečných řad a spigot algoritmů bez Chudnovskyho algoritmu

Překvapivě dobře dopadl Rabinowitz–Wagon algorimus, popsaný v části 3.2.4, pravděpodobně proto, že provádí převážně operace s celými čísly, které jsou rychlejší než operace s čísly s plovoucí čárkou. Ostatní algoritmy dosahují podobných výsledků.

4.3.3 Porovnání výsledků iterativních algoritmů

Na obrázku 4.5 je znázorněna přesnost aproximace podle správných desetinných číslic dosažené za určitý čas v sekundách.



Obrázek 4.5: Porovnání výsledků iterativních algoritmů

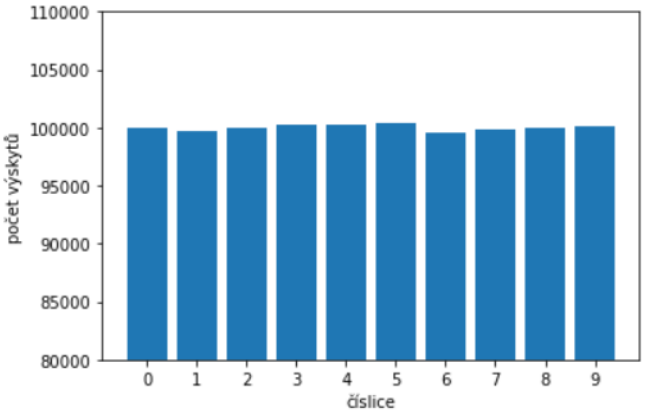
Pro počítání Pí na 10^6 desetinných míst je nejlepší Gauss–Legendreho algoritmus, probraný v části 3.2.1. Tímto algoritmem by to trvalo asi 4,5 sekundy a bylo by potřeba přibližně 9 MB paměti.

Kapitola 5

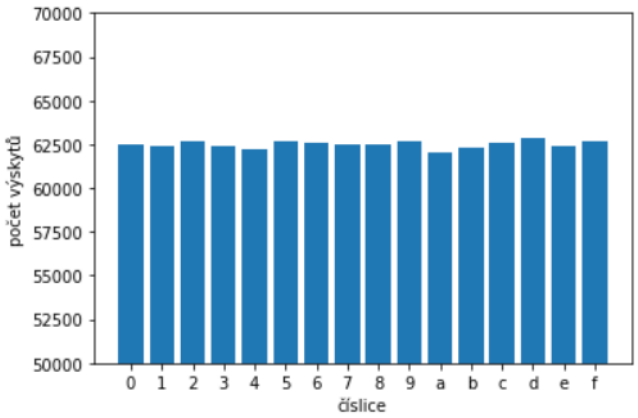
Ověření normality Ludolphova čísla

V této kapitole se budeme zabývat tím, jestli jsou různé aproximace čísla Pí normální čísla. Normalita čísla byla definována v části 1.1. Byly zkoumány aproximace délky 1 milion, 10 milionů, 100 milionů a 1 miliarda číslic a to v desítkové a šestnáctkové soustavě. Aproximace byly staženy jako textové soubory ze stránek [50], [51] a [52]. Byl měřen počet výskytů jednotlivých číslic a počet výskytů všech řetězců délky 2 v obou bázích.

Na obrázcích 5.1 a 5.2 můžeme vidět porovnání počtu výskytů jednotlivých číslic v desítkové a šestnáctkové soustavě v aproximaci o délce 1 milion číslic.



Obrázek 5.1: Počet výskytů číslic v desítkové bázi v 1 milionu číslic



Obrázek 5.2: Počet výskytů číslic v šestnáctkové bázi v 1 milionu číslic

Rozdíly mezi výskyty jednotlivých číslic jsou velmi malé.

V tabulkách 5.1, 5.2, 5.3 a 5.4 jsou minimální a maximální výskyty číslic a řetězců délky 2 v desítkové a šestnáctkové bázi. V tabulkách se také nachází maximální rozdíl od průměrného výskytu a horní odhad pravděpodobnosti výskytu krajních hodnot podle Čebyšovy nerovnosti:

$$P(X \geq \epsilon) \leq \frac{E(X)}{\epsilon}.$$

Počet číslic	Minimum	Maximum	Maximální rozdíl od průměru	Pravděpodobnost výskytu
10^6	99548	100359	452	99.55 %
10^7	999333	1001093	1093	99.89 %
10^8	9993478	10003863	6522	99.93 %
10^9	99986912	100011958	13088	99.99 %

Tabulka 5.1: Statistika výskytu číslic v desítkové bázi

Počet číslic	Minimum	Maximum	Maximální rozdíl od průměru	Pravděpodobnost výskytu
10^6	9721	10239	279	97.29 %
10^7	99314	100816	816	99.19 %
10^8	997874	1002842	2842	99.71 %
10^9	9993549	10008207	8207	99.92 %

Tabulka 5.2: Statistika výskytu řetězců délky 2 v desítkové bázi

Počet číslic	Minimum	Maximum	Maximální rozdíl od průměru	Pravděpodobnost výskytu
10^6	62070	62829	430	99.32 %
10^7	623635	626014	1365	99.78 %
10^8	6246592	6255492	3408	99.95 %
10^9	62482731	62514234	14234	99.98 %

Tabulka 5.3: Statistika výskytu číslic v šestnáctkové bázi

Počet číslic	Minimum	Maximum	Maximální rozdíl od průměru	Pravděpodobnost výskytu
10^6	3729	4093	186.75	95.44 %
10^7	38477	39570	585.5	98.52 %
10^8	388867	392110	1758	99.55 %
10^9	3900347	3911993	5903	99.85 %

Tabulka 5.4: Statistika výskytu řetězců délky 2 v šestnáctkové bázi

Horní odhady pravděpodobností jsou velmi vysoké a se zvyšujícím se množství číslic konvergují ke 100 %.

Z výsledků můžeme usoudit, že aproximace Ludolfova čísla o délce mezi 1 milionem a 1 miliardou číslic jsou normální čísla.

Závěr

Tématem bakalářské práce bylo Ludolfovo číslo, popis nejdůležitějších algoritmů pro jeho počítání, jejich implementace a porovnání jejich rychlostí. Dílčím cílem pak bylo rozhodnout, zda jsou aproximace Pí normální čísla. Algoritmy byly popsány, některé včetně důkazů (například Gregory–Leibnizova řada nebo Chudnovskyho algoritmus).

Algoritmy byly dále implementovány pomocí knihovny GNU-MPFR. Byla změřena rychlost jejich konvergence a pro porovnání výsledků byly algoritmy rozděleny do tří kategorií: jednoduché nekonečné řady, komplexní nekonečné řady a spigot algoritmy a iterativní algoritmy. V neposlední řadě byla zkoumána normalita různých aproximací Ludolfova čísla v různých bázích. Normalita byla zkoumána v desítkové a šestnáctkové bázi pro jednotlivé číslice a řetězce délky 2. Při testování normality vyšly rozdíly mezi výskyty jednotlivých cifer a řetězců velmi malé a bylo rozhodnuto, že zkoumané aproximace Pí jsou normální čísla.

Na tuto práci by bylo možné navzávat přidáním některých algoritmů. Například kvantický a nonický Borwein algoritmus byl v práci jen okrajově zmíněn, ale nebyl podrobně popsán ani naimplementován. Zkoumání normality by bylo možné rozšířit přidáním statistik v jiných bázích a s delšími řetězci.

Literatura

[1] What is pi - Definition and Meaning. *Easy Calculation*, 2017, [online]. [cit. 2020-11-25]. Dostupné z: https://www.easycalculation.com/maths-dictionary/pi.html

[2] Ruščák, M.: ROZHLEDNÍK: Kruh, čtverec a Pí. *Dedeník*, 2017, [online]. [cit. 2020-11-25]. Dostupné z: http://www.dedenik.cz/2017/05/18/rozhlednik-kruh-ctverec-a-pi/

[3] Saipetch, K.: Chronology of Computation of π. *Interesting Things*, 2010, [online]. [cit. 2020-11-26]. Dostupné z: https://kostuff.blogspot.com/2010/08/chronology-of-computation-of.html

[4] Reichl, J.: Matematické kyvadlo. *Encyklopedie fyziky*, 2015, [online]. [cit. 2020-11-26]. Dostupné z: http://fyzika.jreichl.com/main.article/view/205-matematicke-kyvadlo

[5] Roy, R.: The discovery of the series formula for π by Leibniz, Gregory and Nilakantha. *Mathematics Magazine*, ročník 63, č. 5, 1990: s. 291–306.

[6] Reader Reflections. *The Mathematics Teacher*, ročník 108, č. 4, 2014: s. 246–249, ISSN 00255769. Dostupné z: http://www.jstor.org/stable/10.5951/mathteacher.108.4.0246

[7] Rabinowitz, S.; Wagon, S.: A spigot algorithm for the digits of π. *The American mathematical monthly*, ročník 102, č. 3, 1995: s. 195–203.

[8] Fousse, L.; Hanrot, G.; Lefèvre, V.; aj.: MPFR: A multiple-precision binary floating-point library with correct rounding. *ACM Transactions on Mathematical Software (TOMS)*, ročník 33, č. 2, 2007: s. 13–es.

[9] Lischka, C.; Arndt, J.; Lischka, D.; aj.: *Pi - Unleashed*. Springer Berlin Heidelberg, 2012, ISBN 9783642567353. Dostupné z: https://books.google.cz/books?id=MMMPBwAAQBAJ

[10] Bogart, S.: What Is Pi, and How Did It Originate? *Scientific American*, 1999, [online]. [cit. 2020-11-25]. Dostupné z: https://www.scientificamerican.com/article/what-is-pi-and-how-did-it-originate/

[11] Salikhov, V. K.: On the irrationality measure of π. *Russian Mathematical Surveys*, ročník 63, č. 3, Červen 2008, doi:10.1070/RM2008v063n03ABEH004543.

[12] Bernard, S.; Bertot, Y.; Rideau, L.; aj.: Formal proofs of transcendence for e and pi as an application of multivariate and symmetric polynomials. In *Proceedings of the 5th ACM SIGPLAN Conference on Certified Programs and Proofs*, 2016.

[13] Preuss, P.: Are The Digits of Pi Random? Lab Researcher May Hold The Key. *Lawrence Berkeley National Laboratory*, 2001, [online]. [cit. 2020-11-25]. Dostupné z: https://www2.lbl.gov/Science-Articles/Archive/pi-random.html

[14] Eymard, P.; Lafon, J.; Wilson, S.: *The Number π*. American Mathematical Society, 2004, ISBN 9780821832462. Dostupné z: https://books.google.cz/books?id=qZcCSskdtwcC

[15] Brezinski, C.: *History of Continued Fractions and Padé Approximants*. Springer Series in Computational Mathematics, Springer Berlin Heidelberg, 2012, ISBN 9783642581694. Dostupné z: https://books.google.cz/books?id=rxzsCAAAQBAJ

[16] Petrie, F.: *The Wisdom of the Egyptians*. Library of Alexandria, Library of Alexandria, 2020, ISBN 9781613101865. Dostupné z: https://books.google.cz/books?id=s5qGCzMg-nsC

[17] Chaitanya, K.: *A Profile of Indian Culture*. India library, Clarion Books, 1982. Dostupné z: https://books.google.cz/books?id=QJItAAAAMAAJ

[18] Boyer, C. B.; Merzbach, U. C.: *A History of Mathematics*. Wiley, 2011, ISBN 9780470630563. Dostupné z: https://books.google.cz/books?id=bR9HAAAAQBAJ

[19] Azarian, M. K.: Al-Risāla Al-Muhītīyya: A Summary. *Missouri Journal of Mathematical Sciences*, ročník 22, č. 2, 2010, doi:10.35834/mjms/1312233136. Dostupné z: https://doi.org/10.35834/mjms/1312233136

[20] Bailey, D. H.: Some background on kanada's recent pi calculation. *manuscript,(16-May)*, 2003.

[21] Mulican, T.: Calculating Pi: My attempt at breaking the Pi World Record. *Bits and Bytes*, 2020, [online]. [cit. 2020-11-26]. Dostupné z: https://blog.timothymullican.com/calculating-pi-my-attempt-breaking-pi-record

[22] Ramaley, J. F.: Buffon's noodle problem. *The American Mathematical Monthly*, ročník 76, č. 8, 1969: s. 916–918.

[23] Kaiser, G.: *A Friendly Guide to Wavelets*. Modern Birkhäuser Classics, Birkhäuser Boston, 2010, ISBN 9780817681111. Dostupné z: https://books.google.cz/books?id=rfRnrhJwoloC

[24] Bronshtein, I.; Semendiaev, K. A.: *A Guide Book to Mathematics: Fundamental Formulas · Tables · Graphs · Methods*. Springer New York, 2012, ISBN 9781468462883. Dostupné z: https://books.google.cz/books?id=gB3aBwAAQBAJ

[25] Posamentier, A. S.; Lehmann, I.; Hauptman, H. A.: *Pi: A Biography of the World's Most Mysterious Number*. Prometheus, 2013, ISBN 9781615920815. Dostupné z: https://books.google.cz/books?id=PpwBHkxL64kC

[26] Halliday, D.; Resnick, R.; Walker, J.: *Fundamentals of Physics*. Fundamentals of Physics, Wiley, 2013, ISBN 9781118230718. Dostupné z: https://books.google.cz/books?id=HybkAwAAQBAJ

[27] Talwalkar, P.: The Wallis Product Formula For Pi And Its Proof. *MindYourDecisions*, 2016, [online]. [cit. 2021-03-06]. Dostupné z: https://mindyourdecisions.com/blog/2016/10/12/the-wallis-product-formula-for-pi-and-its-proof/

[28] Edwards, C. H. J.: *The Historical Development of the Calculus*. Springer Study Edition, Springer New York, 2012, ISBN 9781461262305. Dostupné z: https://books.google.cz/books?id=ilrlBwAAQBAJ

[29] Borwein, J. M.; Bailey, D. H.; Girgensohn, R.: *Experimentation in Mathematics: Computational Paths to Discovery*. CRC Press, 2004, ISBN 9781439864197. Dostupné z: https://books.google.cz/books?id=10BZDwAAQBAJ

[30] Ayoub, R.: Euler and the zeta function. *The American Mathematical Monthly*, ročník 81, č. 10, 1974: s. 1067–1086.

[31] Beckmann, P.: *A History of Pi*. Griffin Books, St. Martin's Publishing Group, 2015, ISBN 9781466887169. Dostupné z: https://books.google.cz/books?id=10jABQAAQBAJ

[32] Jansson, M.: Approximation of Pi. 2019, [Student Paper].

[33] Shanks, D.; Wrench, J. W.: Calculation of π to 100,000 decimals. *Mathematics of Computation*, ročník 16, č. 77, 1962: s. 76–99.

[34] Craig-Wood, N.: Pi - Chudnovsky. 2011, [online]. [cit. 2020-03-04]. Dostupné z: `https://www.craig-wood.com/nick/articles/pi-chudnovsky/`

[35] Chan, H. H.; Chan, S. H.; Liu, Z.: Domb's numbers and Ramanujan–Sato type series for $1/\pi$. *Advances in Mathematics*, ročník 186, č. 2, 2004: s. 396–410.

[36] Chudnovsky, D. V.; Chudnovsky, G. V.: The computation of classical constants. *Proceedings of the National Academy of Sciences of the United States of America*, ročník 86, č. 21, 1989: s. 8178–8182.

[37] Baruah, N. D.; Berndt, B. C.; Chan, H. H.: Ramanujan's series for $1/\pi$: a survey. *The American Mathematical Monthly*, ročník 116, č. 7, 2009: s. 567–587.

[38] Yee, A.; Kondo, S.: 10 trillion digits of pi: A case study of summing hypergeometric series to high precision on multicore systems. Technická zpráva, 2011.

[39] Yee, A.: 22.4 Trillion Digits of Pi. *Numberworld*, 2016, [online]. [cit. 2021-03-04]. Dostupné z: `http://www.numberworld.org/y-cruncher/records/2016_11_11_pi.txt`

[40] Yee, A.: Google Cloud Topples the Pi Record. *Numberworld*, 2019, [online]. [cit. 2021-03-04]. Dostupné z: `http://www.numberworld.org/blogs/2019_3_14_pi_record/`

[41] Brent, R. P.: Old and new algorithms for pi. *arXiv preprint arXiv:1303.2762*, 2013.

[42] Vestermark, H.: Practical implementation of π Algorithms. *Numerical analyses and methods*, 2016.

[43] Borwein, J. M.; Borwein, P. B.: *Pi and the AGM: a study in the analytic number theory and computational complexity*. Wiley-Interscience, 1987.

[44] Bailey, D. H.; Plouffe, S. M.; Borwein, P. B.; aj.: The quest for pi. *The Mathematical Intelligencer*, ročník 19, 1997: s. 50–56.

[45] Borwein, J. M.; Borwein, P. B.: Ramanujan and pi. *Scientific American*, ročník 258, č. 2, 1988: s. 112–117.

[46] Bailey, D.; Borwein, P.; Plouffe, S.: On the rapid computation of various polylogarithmic constants. *Mathematics of Computation*, ročník 66, č. 218, 1997: s. 903–913.

[47] Bailey, D. H.: The bbp algorithm for pi. Technická zpráva, Lawrence Berkeley National Lab.(LBNL), Berkeley, CA (United States), 2006.

[48] Weisstein, E. W.: BBP-Type formula. *Wolfram*, 2004, [online]. [cit. 2021-03-16]. Dostupné z: `https://mathworld.wolfram.com/BBP-TypeFormula.html`

[49] Heegen, J.: *BBP-numbers and normality*. Dizertační práce, Faculty of Science and Engineering, 2008.

[50] Trueb, P.: All Digits of My Pi World Record. *22.4 trillion digits of pi*, 2017, [online]. [cit. 2021-04-10]. Dostupné z: `https://pi2e.ch/blog/2017/03/10/pi-digits-download/`

[51] Trueb, P.: Various Mathematical Constants to 1 Billion Digits. *Internet Archive*, 2016, [online]. [cit. 2021-04-10]. Dostupné z: `https://archive.org/details/Math_Constants`

[52] Trueb, P.: 1 Billion Hexadecimal Digits of Pi. *Internet Archive*, 2017, [online]. [cit. 2021-04-10]. Dostupné z: `https://archive.org/details/pi_hex_1b`

Příloha A

Zdrojový kód Gregory–Leibnizova algoritmu

```
void GregoryLeibnizAlgorithm::run(mpfr_t destination, int steps)
{
        mpfr_t member;
        mpfr_init2 (member, 100000);
        mpfr_set_d (destination, 0.0, MPFR_RNDD);
        int sign = 1;
        int denom = 1;
        for (int i = 0; i < steps; i++)
        {
                mpfr_set_d (member, 1.0, MPFR_RNDD);
                mpfr_div_si (member, member, denom, MPFR_RNDD);
                mpfr_mul_si (member, member, sign, MPFR_RNDU);
                mpfr_add (destination, destination, member, MPFR_RNDD);
                denom += 2;
                sign = -sign;
        }
        mpfr_mul_si (destination, destination, 4, MPFR_RNDU);
        mpfr_clear (member);
        return;
}
```

Příloha B

Zdrojový kód Gauss–Legendreho algoritmu

```
void GaussLegendreAlgorithm::run(mpfr_t destination, int steps)
{

        mpfr_t a, b, t, p, aOld, tOld, sqrt2;
        mpfr_init2 (a, 10000000);
        mpfr_init2 (b, 10000000);
        mpfr_init2 (t, 10000000);
        mpfr_init2 (p, 10000000);
        mpfr_init2 (aOld, 10000000);
        mpfr_init2 (tOld, 10000000);
        mpfr_init2 (sqrt2, 10000000);
        mpfr_sqrt_ui (sqrt2, 2, MPFR_RNDD);
        mpfr_set_d (destination, 0.0, MPFR_RNDD);
        mpfr_set_d (a, 1.0, MPFR_RNDD); // a_0 = 1
        mpfr_set_d (b, 1.0, MPFR_RNDD);
        mpfr_div (b, b, sqrt2, MPFR_RNDD); // b_0 = 1 / sqrt(2)
        mpfr_set_d (t, 1.0, MPFR_RNDD);
        mpfr_div_si (t, t, 4, MPFR_RNDD); // t_0 = 1/4
        mpfr_set_d (p, 1.0, MPFR_RNDD); // p_0 = 1
        mpfr_set_d (aOld, 1.0, MPFR_RNDD);
        mpfr_set_d (tOld, 1.0, MPFR_RNDD);
        for (int i = 0; i < steps; i++)
        {
                mpfr_add_si (aOld, a, 0, MPFR_RNDD);
                mpfr_add_si (tOld, t, 0, MPFR_RNDD);
                mpfr_add (a, aOld, b, MPFR_RNDD);
                mpfr_div_si (a, a, 2, MPFR_RNDD);
                mpfr_mul (b, aOld, b, MPFR_RNDU);
```

```
                mpfr_sqrt (b, b, MPFR_RNDU);
                mpfr_sub (t, aOld, a, MPFR_RNDD);
                mpfr_mul (t, t, t, MPFR_RNDU);
                mpfr_mul (t, t, p, MPFR_RNDU);
                mpfr_sub (t, tOld, t, MPFR_RNDD);
                mpfr_mul_si (p, p, 2, MPFR_RNDU);
        }
        mpfr_add (destination, a, b, MPFR_RNDD);
        mpfr_mul (destination, destination, destination, MPFR_RNDU);
        mpfr_mul_si (t, t, 4, MPFR_RNDU);
        mpfr_div (destination, destination, t, MPFR_RNDD);
        mpfr_clear (a);
        mpfr_clear (b);
        mpfr_clear (t);
        mpfr_clear (p);
        mpfr_clear (aOld);
        mpfr_clear (tOld);
        mpfr_clear (sqrt2);
        return;
}
```

Příloha C

Zdrojový kód Rabinowitz–Wagonova algoritmu

```
void RabinowitzWagonAlgorithm::run(mpfr_t destination, int steps)
{
        mpfr_t member, multiple;
        mpfr_init2 (member, 100000);
        mpfr_init2 (multiple, 100000);
        mpfr_set_d (destination, 0.0, MPFR_RNDD);
        mpfr_set_d (member, 0.0, MPFR_RNDD);
        mpfr_set_d (multiple, 1, MPFR_RNDD);
        std::vector<int> nums;
        std::vector<int> dens;
        std::vector<int> carries;
        std::vector<int> reminders;
        int len = 10 * steps / 3;
        int entry = 0;
        for (int i = 0; i < len; i++)
        {
                reminders.push_back(2);
                carries.push_back(0);
                nums.push_back(len - 1 - i);
                dens.push_back((len - 1 - i) * 2 + 1);
        }
        for (int i = 0; i < steps; i++)
        {
                mpfr_set_d (member, 0.0, MPFR_RNDD);
                for (int j = 0; j < len - 1; j++)
```

```
                {
                        entry = reminders[j] * 10 + carries[j];
                        reminders[j] = entry % dens[j];
                        carries[j + 1] = entry / dens[j] * nums[j];
                }
                entry = reminders[len - 1] * 10 + carries[len - 1];
                reminders[len - 1] = entry % 10;
                mpfr_add_si (member, member, entry / 10, MPFR_RNDD);
                mpfr_div (member, member, multiple, MPFR_RNDU);
                mpfr_mul_si (multiple, multiple, 10, MPFR_RNDU);
                mpfr_add (destination, destination, member, MPFR_RNDU);
        }
        mpfr_clear (member);
        mpfr_clear (multiple);
        return;
}
```

Příloha D

Seznam použitých zkratek

GNU GNU's not Unix!

MPFR Multiple precision floating-point reliable

BBP Bailey–Borwein–Plouffe

MP Multiple precision arithmetic library

IEEE Institute of electrical and electronics engineers

LGPL Lesser general public license

MB Megabajt

PDF Portable document format

Příloha E

Obsah přiloženého média

- `readme.txt` ..... stručný popis obsahu média
- `src`
  - `pi_algorithms` ..... zdrojové kódy aproximačních algoritmů
  - `normality_testing` ..... zdrojové kódy k testování normality
  - `thesis` ..... zdrojová forma práce ve formátu LaTeX
- `text` ..... text práce
  - `thesis.pdf` ..... text práce ve formátu PDF