VYSOKÁ ŠKOLA FINANČNÍ A SPRÁVNÍ

Fakulta ekonomických studií

Studijní obor: **Aplikovaná informatika**

Bakalářské studium prezenční

Stanislav Kubiš

Analýza po částech spojených užitkových funkcí v reálném herním prostředí

Analysis of piecewise connected utility functions in real game environment

BAKALÁŘSKÁ PRÁCE

Praha 2022

Vedoucí závěrečné práce:

RNDr. Jan Lánský, Ph.D.

Poděkování

Klepněte sem a napište poděkování osobám, které se na práci podílely

Prohlášení

Prohlašuji,

že jsem tuto závěrečnou práci vypracoval/a zcela samostatně a veškerou použitou literaturu a další podkladové materiály, které jsem použil/a, uvádím v seznamu literatury a že svázaná a elektronická podoba práce je shodná. Současně prohlašuji, že souhlasím se zveřejněním této práce podle § 47b zákona č.111/1998Sb., o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších předpisů.

V dne Zvolte datum

Abstrakt

Cílem mé bakalářské práce je analyzovat užitkové funkce u nekonečných her o dvou hráčích s nulovým součtem. Tyto funkce jsou reprezentovány po částech spojenými užitkovými funkcemi na doméně $[0,1] \times [0,1]$. Pro tuto analýze bude navržen algoritmus implementovaný v jazyce MATLAB s kontrolou provedenou jazykem Python. Funkce algoritmu budou vytvářet triangulace a jejich užitkovými funkce. Pro výpočet rovnováhy bude nad triangulacemi vytvořená mřížka, která bude sloužit jako základ pro hru s nulovým součtem. Analýza domény bude spočívat v měření konvergence dílčích výstupů programu.

Abstract

The goal of this bachelor thesis is to analyze utility functions in infinite zero-sum two-player games. These functions are represented by piecewise connected utility functions on $[0,1] \times [0,1]$ domain. Creation of a new algorithm in MATLAB programming language with control made in Python for this analysis. The functions of this algorithm will create triangulations with their utility functions. For evaluating equilibrium, one step of this algorithm creates a grid on such triangulation. This process will serve as a base for a zero-sum game. Basing analysis of the domain on measuring convergence of partial program outputs.

Klíčová slova

Užitkové funkce, lineární programování, teorie her, hry o dvou hráčích, po částech spojené funkce, Nashova rovnováha, hry s nulovým součtem

Keywords

Utility functions, linear programming, game theory, two-player games, piecewise connected functions, Nash equilibrium, zero-sum game

Obsah

1 Úvod

Nejdříve je potřeba zmínit, že tato práce byla pod podobným názvem zpracována mnou v rámci předchozího neúspěšného studia. Práce je vypracována nově a shoda s původní verzí by měla být minimální [3].

Teorie her jako matematická disciplína vznikla až ke konci druhé světové války. U jejího zrodu byli dva muži, matematik John von Neumann a ekonom Oskar Morgenstern [1]. Společně dali základ metodám, které dosahují v současné době velkému praktickému využití. To je dáno neustále se vylepšující počítačovou infrastrukturou. V praxi je nalezneme v metodikách kybernetické bezpečnosti, umělé inteligenci, ale i v ekonomických oblastech. Jednou ze zmíněných metod je teorie ohledně her hraných dvěma hráči, v práci probereme její verzi s nulovým součtem postavené na hledání rovnováhy hráčů. Pojmenována je po svém objeviteli Johnu Forbes Nashovi jako Nashova rovnováha [4]. Ta leží uprostřed teorie, která podtrhuje to, čeho se snažíme dosáhnout. Experiment v této práci je prováděn na omezené doméně o délce jedna, na které budou tvořeny po částech spojené funkce. Je nutné podotknout, že složitost hledání Nashovy rovnováhy se odvíjí od velikosti dané hry. Náš případ pokrývá potenciálně nekonečný herní prostor. Přestože se v přirozeně vyskytujících situacích jedná o nepočitatelné strategické prostory, tak budou předvedeny teoretické prvky s definicemi uvedenými na konečně velkých hrách. To způsobí, že namísto smíšené strategie dosahujeme u výsledku spíše pravděpodobnostnímu rozdělení.

V původní práci byly všechny funkce označené jako afinní, přestože se někdy jednalo pouze o lineární. To je dáno tím, že některé funkce měly nulový ofset. Podle definice se jako afinní mohou označovat, ale též odpovídají lineárním, které ofset nemají. To je důvod, proč je zadání pojmenováno jako spojené funkce a není omezeno pouze na funkce afinní.

Celkově nebyla problematika na tomto prostředí moc studována. V práci bude vysvětlena a analyzována na praktické simulaci. Cílem je totiž ukázat, zda nekonečné hry s nulovým součtem o dvou hráčích a po částech spojenými užitkovými funkcemi dosahují ustálené rovnováhy. Úkolem je nasimulovat herní prostředí a analyzovat dílčí výstupy, zda konvergují k nějakým hodnotám. K tomu je potřeba se nejprve seznámit s teoretickými prvky problematiky strategických her pro dva hráče, a zvláště dávat pozor na algoritmus počítající Nashovu rovnováhu. Poté bude naprogramován algoritmus simulující strategický herní prostor pro oba hráče.

Algoritmus je založený na náhodné generaci bodů spolu s hodnotami jejich užitkových funkcí podél mřížky se specifikovanou jemností. Mezi těmito body budou vytvořeny triangulace, kde každá úsečka reprezentuje afinní, případně lineární funkci při nulovém bodovém užitku. Z těchto bodů jsou vyvedeny přímky pro doplnění užitkové matice pomocí průsečíků s úsečkami triangulace. Jelikož je cílem simulovat potenciálně nekonečnou hru, je pro zvětšení použit iterační postup hledající další průsečíky z nově nalezených bodů. Jednotlivá vyhodnocování jsou dělána pomocí duálního vzorce lineárního programování pro hru o dvou hráčích.

Díky zkušenostem nabitým předchozí prací bude program naimplementován v matematicky orientovaném jazyce MATLAB. Pro ověření pozorovaných výsledků budou dílčí výsledky kontrolovány výpočty v jazyce Python. Jako knihovny budou na triangulaci využity jmenovitě následující tři a to `delaunay`, `triangulation`, a `delaunayTriangulation`. Jejich výstupy budou porovnány a nejlepší z nich bude využita. Pro vyhodnocení lineárního programování bude použit `linprog`, který by měl poskytnout velmi přesné výsledky oproti jeho interpretacím v jazyce Python.

Po provedení algoritmu budeme pozorovat výslednou hodnotu strategického prostoru hry a podle toho bude možno usoudit, zda její hodnota někam konverguje. Tato myšlenka je založena na Glicksbergovu teorému pojímajícím existenci Nashovy rovnováhy ve všech kontinuálních hrách [2].

2 Teorie her

Teorie her zahrnuje několik nástrojů uzpůsobených k porozumění fenoménu interakce pozorované mezi rozhodujícími, kterým budeme říkat hráči nebo agenti. Základním kamenem teorie je předpoklad racionality účastníků dané hry společně s jejich očekáváními a vědomostmi, jelikož uvažuje jejich strategické motivy. Hlavní myšlenkou modelů teorie je reprezentovat reálné situace velmi abstraktně. Díky tomu je využitelnost těchto modelů možná pro mnohé fenomény a pro jejich studium. Třeba studium politické soutěže je jedním z využití Nashovy rovnováhy. Vysvětlení délky včelího jazyka je naopak výsledkem využití smíšených strategií. Mezi aplikovanou a čistou teorií je hranice vágní. Tomu dopomáhá i používání reálných aplikací pro čistě teoretické studium. [5]

2.1 Evoluce teorie

V novodobé historii se potýkáme se vznikem konceptu pravděpodobnosti, která tvoří základní prvek principu smíšené pravděpodobnosti. Začátek pravděpodobnostního kalkulu se datuje k polovině 17. století matematiky Fermatovi a Pascalovi. Na tyto matematiky navázala rodina Waldegravova před 320 lety, kdy James Waldegrave popsal první řešení maticové hry pomocí smíšené strategie [13].

Po více než stoleté mezeře izolovaných příkladů zveřejňuje Émilie Borel na počátku dvacátého století své poznámky o symetrických hrách o dvou hráčích s nulovým součtem a konečným počtem hráčů. Popisoval případy s existující čistou strategií pro každého z nich. Po přibližně třiceti letech od tohoto objevu přichází doba von Neumanna a Oskara Morgesterna. Ti uvedli v roce 1944 knihu Teorie her a ekonomické chování. Od detailního popisu teorie v knize též zkoumají její rozsáhlé užití v mnoha různých odvětvích. Pět let poté přichází John Forbes Nash se svou teorií rovnováhy ve hrách. Nyní se jedná o nejvíce ovlivňující formulaci teorie her [13].

V roce 1954 přichází Američané Shapley a Shubik s aplikací teorie do politických věd. Svůj příklad kooperativních her využili na určení moci jednotlivých států Spojených Národů. Další rozšíření přišlo o pár let později v oblasti evoluční biologie. Vědci zde zkoumali vše od kooperace po porozumění konfliktů mezi zvířaty a rostlinami. O 30 let později přichází Robert Axelrod s teorií kooperace zkoumající její reciprocity i mezi skupinou egoisticky smýšlejících lidí [13].

Historické shrnutí uzavřu informací o zisku Nobelovy ceny za ekonomii matematika Nash spolu s vědci Harsanyim a Seltenem za jejich kontribuci do nekooperativní teorie her využívané v matematice [13].

2.2 Hry s nulovým součtem

Než budou zmíněné hry s nulovým součtem, je potřeba vysvětlit hry v normálové formě. V herní teorii jde o reprezentaci strategické interakce hráčů. Jedná se o reprezentaci užitku pro každý stav světa každého hráče, a to ve speciálním případě, kdy pouze kombinované akce hráčů ovlivňují stav světa. Tento speciální případ se může zdát nezajímavý k uvažování. Přesto se ukáže, že náhodnost v prostředí má vliv na redukci stavů světa ve hře v normální formě. Nesmíme též opomenout existenci ostatních redukovatelných her. Pro příklad třeba existují takové, které používají čas jako element. Ani tento fakt nemění nic na tom, že reprezentace v normálové formě je v herní teorii jako jedna z nejvíce zásadních formulací [4].

Definice hry v normálové formě. *Konečná hra o n hráčích v normálové formě je seřazený seznam prvků (N, A, u), kde:*

N značí konečný počet hráčů n indexovaných pomocí i

$A = A_1 \times \ldots \times A_n$, kde A_i značí konečný set akcí k dispozici hráči i. Každý vektor $a = (a_1, \ldots, a_n)$ z A se nazývá profil akcí

$u = (u_1, \ldots, u_n)$ kde $u_i: A \rightarrow R$ je užitková funkce s reálnou hodnotou hráče i [4]

Pro tento typ her se n-dimenzionální matice využívají na jejich reprezentaci. Úvodní příklad takovéto hry s názvem vězňovo dilema je reprezentován ve dvou dimenzích. Pro porozumění takové maticové hry platí, že každý sloupec reprezentuje možné hodnoty jednoho hráče, zatímco řádek toho druhého. Každá buňka odpovídá možnému výsledku hry. Hodnoty v nich odpovídají užitkům hráčů tak, že první hodnota odpovídá řádkovému a druhá sloupcovému hráči [4].

	K	*P*
K	*a,a*	*b,c*
P	*c,b*	*d,d*

Obrázek 1: Užitková matice Vězňova dilematu [4], přeloženo do češtiny

Vězňovo dilema je definované pro každou instanci $c > a > d > b$ [4].

Naše hry s nulovým součtem patří do normálových her, přesněji her s konstantním součtem. Ještě, než se k nim přesuneme, vysvětlíme si hry se společným užitkem. Jedná se o specifické verze normálových her se speciálním omezením. Hlavní rozdíl oproti hrám typu vězňova dilematu je ten, že každá buňka je reprezentována pouze jednou hodnotou pro oba hráče. Formálně jsou definovány následovně [4].

Definice hry se společným užitkem. *Hra se společným užitkem je hra, ve které všechny profily akcí* a *z* $A_1 \times \ldots \times A_n$ *a všechny páry agentů* i*,* j *platí* $u_i(a) = u_j(a)$ *[4].*

Čistě koordinační či týmové hry je též používaný název pro tyto hry. Důvodem je fakt nekonfliktnosti zájmů agentů v nich. Maximální benefit je společnou koordinační výzvou těchto zainteresovaných agentů [4].

Nyní se můžeme posunout ke hrám s nulovým součtem. Oproti hrám se společným užitkem jsou hlavně využívány v situacích se dvěma hráči. Jak již bylo zmíněno tento typ her je ve skupině těch s konstantním součtem [4].

Definice her s konstantním součtem. *Hra v normálové formě o dvou hráčích má konstantní součet, pokud existuje konstanta* c *taková, že pro každý strategický profil* a *z* $A_1 \times A_2$ *se jedná o případ* $u_1(a) + u_2(a) = c$ *[4].*

Již bylo zmíněno, že v simulaci bude hra typu s nulovým součtem. Bude tedy platit, že má konstanta c hodnotu nula. Situace ve hrách s nulovým součtem jsou reprezentovány čistou soutěží oproti takzvané čisté spolupráci jako tomu bylo u her se společným užitkem. Čistá soutěž označuje situaci, kdy při zisku jednoho dochází ke ztrátě druhého. Důvod pro hlavní využití v situacích s pouze dvěma hráči je ten, že přidáním dalšího je umožněno připojení hloupého hráče pro dotvoření nulového součtu užitků. Jelikož tento hráč neovlivňuje užitky ostatních pomocích svých akcí, tak mohou být užitky původních dvou i nekonečně velké [4].

Mezi základní příklady her o nulovém součtu patří panna nebo orel, či stříhání. V prvním případě se jedná o hru dvou hráčů, kde oba mají svou vlastní minci se stranami pojmenovanými panna a orel. Jakmile dojde na hru, každý zvolí jednu stranu mince. Pokud oba zvolí stejnou hodnotu dostane obě mince první, v opačném případě ten druhý.

	P	*O*
P	*1,-1*	*-1,1*
O	*-1,1*	*1,-1*

Obrázek 2: Užitková matice pro hru panna nebo orel [33], přeloženo do češtiny

Příklad stříhání rozšiřuje tento příklad o další strategii. Existují zde tři možnosti kámen, nůžky, a papír. Součet užitků je konstantně roven nule. Při shodě je užitek obou je roven nule, v ostatních vyhrává jeden z nich a výhra je označená hodnotou jedna [4].

	K	P	N
K	0,0	-1,1	1,-1
P	1,-1	0,0	-1,1
N	-1,1	1,-1	0,0

Obrázek 3: Užitková matice pro hru kámen, nůžky, papír [34], opraveno a přeloženo do češtiny

2.3 Herní strategie

V definici her v normálové formě jsou uvedeny tři základní proměnné. Jedná se o počet hráčů, dostupné akce, a užitkové funkce. S akcemi jsme již pracovali a nyní se podíváme na to, jak si je hráči volí. Této volbě se v herní teorii říká strategie. Zmíníme dva hlavní typy strategií, a to čisté a smíšené. Pokud má hráč k dispozici několik akcí a rozhodne se vybírat si právě jednu, pak se jedná o čistou strategii. Profil čisté strategie je název výběru agentovi čisté strategie. Další možností je náhodně vybírat napříč dostupnými akcemi. Tento výběr je ovlivněn nějakou pravděpodobnostní distribucí a nazývá se strategií smíšenou [4].

Důvodem uvádět tyto dvě strategie společně je fakt, že speciálním případem smíšené strategie je čistá strategie. Koncept smíšeného strategického profilu pojednává o pravděpodobnostním rozdělení hraní jednotlivých akcí. Tuto myšlenku lze ukázat na několika příkladech. Zde je popsána situace, v již zmíněné hře panna nebo orel. Smíšená strategie je zde pravděpodobnost volby. Označme pravděpodobnost sigma jako hodnotu mezi 0 a 1, matematicky značeno jako $\sigma \in (0,1)$. Hodnota σ je rovna pravděpodobnosti volby panny. Z toho vyplývá, že $1 - \sigma$ je pravděpodobnost volby orla. Daný vzorec dává najevo společnou hodnotu obou jevů rovnou jedné. Hodnota pro jednu stranu může být jakákoli od nuly do jedné [10]. Formální zápis je následující:

$$\Delta S_i = \{(\sigma(H), \sigma(T)): \sigma(H) \geq 0, \sigma(T) \geq 0, \sigma(H) + \sigma(T) = 1\} \text{ (1) [10]}$$

Zápis vzorce (1) označuje set smíšených strategií, ve kterých je součet pravděpodobností roven jedné a jsou obě nenulové. Podobný případ se třemi možnostmi existuje například ve hře kámen, nůžky, papír. Ve hře panna nebo orel je jednou takovou kombinací rovnost pravděpodobnosti pro pannu jedna a nula pro orla [10].

2.4 Nashova rovnováha

Hlavní myšlenka Nashovy rovnováhy spočívá v pohledu na hru z hráčské perspektivy namísto vnějšího pozorovatele. Představme si, že hráč je obeznámen s tím, jak budou ostatní hrát. Tehdy pouze stačí vybrat nejlepší strategii, jelikož by se jednalo pouze o hru s jedním hráčem. Formálně můžeme definovat hráčův strategický profil s. Díky tomuto profilu lze definovat jeho nejlepší odezvu [4].

Definice nejlepší odezva*. Nejlepší odezvou hráče i na strategický profil s_{-i} je smíšená strategie $s_i^* \in S_i$ taková, že $u_i(s_i^*, s_{-i}) \geq u_i(s_i, s_{-i})$ pro všechny strategie s_i z S_i [4].*

Obecně neplatí, že by všechny nejlepší odezvy byly unikátní. Většinou se jedná o jejich nekonečný počet a pouze v čisté strategii dostáváme jedinou nejlepší odezvu. Když strategický profil obsahuje více než dvě akce, je potřeba, aby hráč neměl žádnou preferenci mezi nimi. V opačném případě by mohl redukovat na nulovou hodnotu pravděpodobnost hraní nějaké z ostatních akcí. Díky tomu je nejlepší odezva též každá kombinace z těchto akcí. Podobně je tomu u individuálně nejlepších odezev u dvou čistých strategií. V typické takové hře neplatí, že by byla strategie oponujících známá. Bohužel vzhledem k tomu není odezva konceptem řešení, jelikož neidentifikuje pro obecná řešení žádné zajímavé výsledky. Každopádně pro vytvoření definice Nashovy rovnováhy, jakožto centrálního prvku herní teorie, je možné využít nápadu nejlepší odezvy [4].

Definice Nashovy rovnováhy*. Strategický profil $s = (s_1, \dots, s_n)$ je Nashova rovnováha, pokud pro všechny agenty i, s_i je nejlepší odezva k s_{-i} [4].*

Výhoda rovnováhy spočívá v její stabilitě. Informace ohledně strategie protivníka nemá vliv na změnu hráčovi vlastní strategie. Jako dva druhy této rovnováhy jsou slabý a striktní Nash podle toho, zda se jedná o unikátně nejlepší odezvu nebo nikoli [4].

Definice striktní Nash. *Strategický profil $s = (s_1, \dots, s_n)$ je striktní, pokud pro všechny hráče i a všechny strategie $s_i' \neq s_i$ je $u_i(s_i, s_{-i}) > u_i(s_i', s_{-i})$ [4].*

Definice slabý Nash. *Strategický profil $s = (s_1, \dots, s_n)$ je slabý, pokud pro všechny hráče i a všechny strategie $s_i' \neq s_i$ je $u_i(s_i, s_{-i}) \geq u_i(s_i', s_{-i})$ a s není striktní Nashova rovnováha [4].*

Z definic lze vidět, že striktní Nashova rovnováha je stabilnější. Ve slabém Nashovi se totiž nejedná o protivníkovu rovnovážnou strategii. Z toho důvodu jsou smíšené strategie slabými, zatímco čistá strategie může v závislosti na hře být obojím [4].

2.5 Konečné a spojité hry

V této kapitole budou uvedeny dva odlišné typy her. Jedná se o hry konečné a kontinuální. Jak bude objasněno, tak v této bakalářské práci se jedná o kontinuální typ.

První zmíněná kategorie her se rozděluje na konečné a nekonečné. Smysl konečné hry je v tom, že na jejím konci je výhra. Oproti tomu nekonečné hry jsou hrány se záměrem pokračování ve hraní [6].

Definice konečných normálových her. *Konečná normálová hra je hra v normálové formě, kde všechny sety akcí $A_1, A_2, \dots, A_n$ jsou konečné [22].*

Jelikož tento typ her požaduje výhru, dostane se ke svému konci právě tehdy, když někdo vyhraje. Tomu se stane, jakmile se hráči dohodnou na jednom výherci. Pro jeho určení není určena žádná další podmínka. Problém v jeho určení může nastat v případě, kdy pozorovatel nebo rozhodčí nesouhlasí s dohodnutým výsledkem. Hráči přestávají hrát a nemohou být nikým přemluveni k pokračování ve hraní. To je dáno neexistencí konečné hry, ve které by mohli být hráči nuceni ke hře. Tento princip také spojuje konečný a nekonečný typ her. Svobodné hraní je jediné akceptovatelné [6].

Tyto hry potřebují k jasnému konci také jasný začátek. Na jejich začátku jsou dány nějaké, hráči předem domluvené, vytyčené hranice. S těmito omezeními souvisí i ta početní a prostorová. Tím se myslí jasný počet hráčů společně s jasně určeným hracím polem. Pro příklad může být uvedena druhá světová válka, kde bylo Švýcarsko určeno jako neutrální a obě strany si určili nebombardovatelné město. Dalšími definovanými prvky je čas hry a příslušnost jednotlivých stran. V těchto hrách je totiž pouze jeden výherce složený z jednoho hráče nebo týmu. Jako další příklady těchto her jdou uvést piškvorky, nebo dáma [6].

Ve druhém případě se jedná o hry nekonečné. Tyto hry mají s těmi konečnými společné pouze jedno již zmíněné pravidlo, a to o dobrovolnosti hraní. V ostatních případech se jedná o naprosto opačné herní případy. Tato hra neřeší počet hráčů, kde hrají, nebo v jakém časovém úseku. Vzhledem k těmto faktům není možno určit konec těchto her. Velkou výhodou nekonečným her je možnost v nich hrát hry konečné [6].

Předchozí zmíněné konečné neboli diskrétní jsou hlavně definovány svou omezeností v počtech hráčů, možností a ostatních prvků. Kontinuální hry tento koncept rozvíjí o možnost nekonečného počtu čistých strategií. Díky tomu může být zahrnuta možnost omezeného intervalu obsahujícího čisté strategie [7].

Definice kontinuální hry. *Kontinuální hra je hra* $\langle I, (S_i), (u_i) \rangle$ *kde* I *je konečný set,* S_i *je neprázdný kompaktní metrický prostor, a* $u_i : S \rightarrow R$ *jsou kontinuální funkce [7].*

Obecné matematické struktury nekonečných setů jsou kompaktní metrické prostory dobře aproximovatelné pomocí konečných setů. Důležitým faktem v tomto prostoru je konvergence nekonečných sekvencí. Konečně dimenzionální Euklidův prostor a každá jeho podmnožina je kompaktní metrický prostor. Specifičtěji lze říct, že tento fakt platí vždy, pokud pro vzdálenost bodů x a y platí $|x - y|$ na libovolném omezeném intervalu [7].

Nyní se dostáváme ke Glicksbergovu teorému pro Nashovu rovnováhu.

Glicksbergův teorém: *Každá kontinuální hra má alespoň jednu Nashovu rovnováhu [7].*

Smíšená strategie v našem kontinuálním strategickém prostoru může být nekonečně dimenzionální. Je tedy potřeba využít jinou než Kakutaniho verzi teorému, a to více silný teorém pro pevný bod. Můžeme použít alternativu naší verze, která se zakládá na následujících bodech:

- Aproximace pomocí konečné hry, která lépe odpovídá diskretizaci originální hry.
- Pomocí Nashova teorému je možno pro všechny aproximace vyhledat jeho rovnováhu.
- Za užití předpokladu kontinuity a slabé topologie je konvergence k rovnováze původní hry následně ukazatelná [7].

Pro potvrzení toho, že náš případ zahrnuje právě kontinuální hry poukážeme i na nekontinuální verzi. Tento typ je využíván v případech, kdy se nejedná o kontinuální užitkové funkce. Mezi příklady patří soutěžní modely, nebo modely přetížené soutěže [7].

2.6 Užitkové funkce

Nyní je potřeba vysvětlit, k čemu se zmíněné užitkové funkce využívají a jak fungují. Teorie užitku je využívána k modelování agentových zájmů. Jedná se o dominantní způsob tohoto modelování. Tato teorie zahrnuje kvantifikování preferencí agenta napříč dostupnými alternativami. Spolu s tím slouží k popisu situací agentovi nejistoty. Ta nastává, jelikož si nemůže být jistý tím, jakou alternativu obdrží. Užitková funkce slouží k mapování hodnot herního světa do těch reálných. Tyto hodnoty odpovídají agentově spokojenosti v daném herním stavu. Očekávaná hodnota užitkové funkce je definice užitku v situacích hráčské nejistoty ohledně stavu světa, ve kterém se nachází. Hodnota je určována s respektem na stavy a jejich pravděpodobnostní rozdělení [4].

V situacích, kdy má agent dané užitkové funkce je rozumné předpokládat racionální uvažování i přes nejistotu herního prostředí. To platí v případech, kdy jsou dostatečně agentem reprezentovány a známy výstupy s jejich pravděpodobnostmi. Pro situace s alespoň dvěma hráči maximalizujícími svůj užitek v daném herním prostředí se tento koncept stává velmi komplikovaným a k jeho studiu jsou využívány nekooperační hry [4].

Pojem nekooperační není úplně ideální, jelikož svým názvem může mást. Nejedná se totiž pouze o situace, kde jsou zájmy pro odlišné agenty v konfliktu. Tento obor se zajímá hlavně takovými případy, avšak není tomu tak ve všech situacích. Podobně jsou na tom komplementární koaliční neboli kooperativní hry, které nezahrnují pouze situace, kde jsou zájmy jednotné pro různé agenty. Základní rozdíl mezi nimi je modelování agentů. Zatímco kooperativní hry zkoumají všechny jako jednu skupinu, tak nekooperační hry zkoumají každého jednotlivě. Nejznámějším příkladem nekooperačních her je již jmenovaný příklad Vězňova dilematu [4].

Nyní lze vypozorovat, že pro výsledek roven nule je potřeba opačným hodnotám našich dvou hráčů. Jejich užitky je možno reprezentovat jako minimální, respektive maximální hodnotu. Tato metodika se nazývá minmax, případně maxmin, reprezentující hodnoty užitků vyhledávaných jednotlivými hráči [8].

Definice minmax pro dva hráče. *Ve hře o dvou hráčích je minmax strategie pro hráče i proti hráči $-i$ hodnota* $\arg min_{si} max_{s-i}\, u_{-i}(s_i, s_{-i})$ *a hráče $-i$ minmax hodnota je* $min_{si} max_{s-i}\, u_{-i}(s_i, s_{-i})$ *[8].*

V případě maxmin se jedná o prohození argumentů min a max s výsledkem $max_{si} min_{-si}$. Tento koncept může zprvu dávat smysl jen při souběžném hraní, avšak může být díky následující temporální intuici lépe porozuměn. V maxmin případě si lze představit, že nejprve hráč i musí vybrat vybere nejlepší možnost potenciálně smíšené strategie. Teprve poté budou ostatní hráči vybírat na základně tohoto pozorování strategii minimalizující jeho očekávaný užitek. Nelze očekávat, že hráči budou zainteresováni pouze v poškození hráče i. Díky tomu bude hodnota obdržená rovna alespoň užitku akce, kterou hráč vybral pomocí své maxmin strategie. Tento fakt poukazuje na rozumnost dané volby pro konzervativní hráče. Ti hrají bez ohledu na ostatní hráče a pouze se soustředí na svůj vlastní zájem o maximalizaci svého očekávaného užitku. Zmíněné strategie jsou si komplementární. Když se opřeme o definici lze vidět, že se při strategii snaží hráč i udělat volbu minimalizující maximální možný užitek hráče $-i$. Opačně se hráč $-i$ snaží maximalizovat minimální možný užitek hráče i. To znamená, že jeden hráč může uškodit druhému bez ohledu na svůj vlastní užitek. Děje se tomu tak v opakovaných hrách [8].

Minmax teorém. *V jakékoli konečné hře o dvou hráčích s nulovým součtem platí pro každou Nashovu rovnováhu užitkový zisk o hodnotě rovnou oběma maxmin a minmax hodnotám [8].*

Tato metodika demonstruje rovnost minmax a maxmin strategií pro hry o dvou hráčích ve vztahu k Nashově rovnováze. Díky zmíněnému teorému platí pro hry s nulovým součtem o dvou hráčích následující:

1. *Maxmin hodnota každého hráče je rovná jeho minmax hodnotě. Konvence určuje maxmin hodnotu hráče jedna jako hodnotu celé hry.*
2. *Pro oba hráče je set maxmin strategií shodný se setem jejich minmax strategií.*
3. *Jakýkoli maxmin strategický profil (nebo rovněž také minmax strategický profil) je Nashovou rovnováhou. Dále lze říct, že jsou všechny Nashovou rovnováhou. Díky tomu mají všechny Nashovy rovnováhy shodné užitkové vektory (jmenovitě ty, ve kterých získá hráč jedna hodnotu hry) [8].*

Pro hry o dvou hráčích s nulovým součtem je Nashovou rovnováhou sedlový bod. Pro něj platí, že při jakékoli změně jednoho hráče si ten druhý polepší. K výpočtu je třeba reprezentovat hru pomocí lineárního programování. Doba výpočtu je poté možná v polynomiálním čase. Formální zápis hry s nulovým součtem o dvou hráčích je hra $H = (\{1,2\}, A_1 \times A_2, (u_1, u_2))$. Prostor všech dostupných možností hráčů je kartézský součin akcí $A_1 \times A_2$. Očekávaný užitek hráče i bude označován jako U_i^*, též hodnota hry z předchozí definice. Vzhledem k tomu, že se jedná o nulový součet užitků je potřeba docílit nulového součtu, tedy $U_1^* = -U_2^*$. Nyní se můžeme opět opřít o teorém, který nám říká, že hodnota U_1^* je konstantní rovnováha a jiného výsledku není dosaženo ani při použití minmax strategie hráčem 2. Pomocí toho dostáváme následující lineární program [8]:

$$\min U_1^*$$

$$za\ podmínek \sum_{k \in A_2} u_1(a_1^j, a_2^k) * s_2^k \leq U_1^* \quad \forall j \in A_1$$

$$\sum_{k \in A_2} s_2^k = 1$$

$$s_2^k \geq 0 \quad \forall k \in A_2 \text{ (2) [8]}$$

Zatímco smíšené strategie $s._2$ a U_1^* jsou proměnné, tak užitky $u_1(.)$ zůstávají konstantní. Pojďme si vysvětlit, co nám jednotlivé podmínky říkají. V první podmínce jsou čisté strategie hráče jedna reprezentovány a_1^j. Ty nám značí, že při hraní jakékoli akce j z A_1 je jeho očekávaný užitek při smíšené strategii hráče 2 značené s_2^k nejvýše U_1^*. V setu nejlepších odezev hráče 1 budou pouze takové čisté strategie s očekávaným užitkem rovným hodnotě U_1^*, na rozdíl od ostatních s očekávaným užitkem směřujícím k nižším hodnotám. Nesmíme zapomenout na to, že U_1^* je proměnná. Ta vybere smíšenou strategii hráče 2 v lineárním programu tak, aby U_1^* vzhledem k omezením mělo co nejmenší hodnotu. Nyní je tedy zřejmé, že první dva řádky programu označují smíšenou strategii hranou hráčem 2 s cílem minimalizace užitku protihráče hrajícího svou nejlepší odezvu. Přesně tak tomu je v definici. Ostatní podmínky zajištují konzistenci pravděpodobnosti reprezentované proměnnou s_2^k. Přesněji její pravidla, že součty všech jednotlivých pravděpodobností jsou rovny jedné, a že neexistuje záporná pravděpodobnost [8].

Vysvětlený lineární program nám dává rovnovážnou smíšenou strategii hráče 2, který se snaží minimalizovat svůj užitek. Podobným způsobem lze pro smíšenou strategii hráče 1 vytvořit pomocí duality lineární program. Dualita funguje pomocí prohození hráčů, otočení pravidel nerovnosti a obrácení minimalizace. Pro druhého hráče je tedy cílem maximalizovat hodnotu U_1^* vzhledem k touze hráče 1 o maximalizaci svého užitku. Duální forma k předchozímu lineárnímu programu poté odpovídá:

$$\max U_1^*$$

$$za\ podmínek \sum_{j \in A_1} u_1(a_1^j, a_2^k) * s_1^j \geq U_1^* \quad \forall k \in A_2$$

$$\sum_{j \in A_1} s_1^j = 1$$

$$s_1^j \geq 0 \quad \forall j \in A_1 \text{ (3) [8]}$$

Když se podíváme na omezení v tomto duálním programu vidíme, že podmínky pro pravděpodobnost zůstávají stejné. Oproti hráči 1 nám zde první nerovnost určuje očekávanou hodnotu užitku. Změna oproti původnímu programu nastává v hodnotě proměnné s_1^j značící smíšenou strategii hráče 1. Celkově nám nerovnost značí, že při hraní libovolné akce bude užitek roven alespoň U_1^*.

3 Po částech spojené užitkové funkce

Než si ukážeme po částech spojené funkce je třeba uvést prvky, ze kterých jsou složené. Jmenovitě se v simulaci bude jednat o lineární a afinní funkce. Ty svými výškovými hodnotami budou reprezentovat užitkové funkce.

Tyto dva typy funkcí se liší ve svém tvaru. Afinní funkce mají navíc oproti lineárním přičtený ofset. Dalo by se říct, že lineární funkce je afinní funkce s ofsetem rovným nule. Obě funkce reprezentují funkce s doménami na podmnožině s reálnými hodnotami [9]. Jelikož je afinní funkce nadmnožinou té lineární, bude uvedena formální definice pouze té.

Definice afinní funkce. *Funkce* $A: R_m \to R_n$ *je afinní, jestli existuje spojení lineární funkce* $L: R_m \to R_n$ *a vektoru b v* R_n *takové, že:*

$$A(x) = L(x) + b$$

Pro všechna x *v* R_m *[9].*

Zobrazení $R_m \to R_n$ označuje skupinu funkcí, pro které je velikost domény podmnožinou R_m a rozsah je podmnožinou R_n [9]. Označení v definici písmenem A označuje funkci afinní a L funkci lineární.

3.1 Složitost hledání rovnováhy

Při rozebírání časové složitosti problémů se zkoumají dvě hlavní skupiny. Jedná se o polynomiální neboli řešitelné v polynomiálním čase a NP-těžké problémy. V předchozích kapitolách byly ukázány příklady na hry, ve kterých je možnost odhadovat hráčské chování. Čistá strategie může být jedna z odhadovaných Nashových rovnováh. Ta se skládá z nejlepší reakce hráče na situaci, v příkladu Vězňova dilematu se jedná o přiznání. Oproti tomu příklady jako kámen nůžky papír nemají jednoznačnou čistou strategii. V daných případech se jedná o smíšenou rovnováhu reprezentovanou pravděpodobnostním rozdělení každého hráče [12].

Díky výzkumu Johna Nashe víme o existenci smíšené rovnováhy ve všech hrách. Tedy pro každou akci má hráč dostupnou nejlepší reakci. Jedná se o fakt vycházející z informace o existenci rovnováhy. Nejlepší reakce ovšem nemusí být vždy dobře a efektivně spočítatelná. Nelze předpokládat uskutečnitelnost problémů, kde výpočet rovnováhy trvá dobu existence několika vesmírů. Pro hry s nulovým součtem objevil George Dantzig ve 40. letech možnost výpočtu pomocí lineárního programování, který byl uveden v předchozí kapitole. Tento algoritmus nelze využít pro hry s nenulovým součtem. Mnoho takových problémů má polynomiální časovou složitost, avšak se nejedná o pravidlo pro všechny takové. Pro ostatní problémy neexistuje jasný postup, a tedy se jedná o problémy NP-úplné. Zmíněná úplnost podtrhuje neefektivitu počítání takových problémů, pokud neplatí polynomiální $P = NP$. Zmíněná rovnice není dosud prokázána a matematici se dělí na skupiny věřících a nevěřících. Obecně lze říct, že velikost problému ovlivní jeho časovou složitost i v případech již známého algoritmu [12].

3.2 Použití her s nulovým součtem

Hry s nulovým součtem dostávají rozsáhlého využití. Není to dáno pouze novodobím vývojem moderních technologií. Již u počátků novodobé teorie her zkoumali von Neumann a Morgenstern využití v ekonomicky propojených případech [14]. Mimo jiné též již zmíněná kniha o Ekonomickém chování, jejíž vliv je pozorovatelný dodnes. Kniha poskytla nový rámec pro analyzování strategického rozhodovaní. Její podstata je vyvoditelná z množství nově napsaných vysokoškolských učebnic v posledních letech [15]. Dá se předpokládat, že se jedná o hlavní využití teorie her.

Kromě kategorie ekonomické se též jedná o využití v biologii a jeho zkoumání pravděpodobnosti vyhynutí jednotlivého druhu. Dalším užitím jsou sportovní utkání při analýze chování hráčů během utkání, nebo ve vojenském strategizování [21].

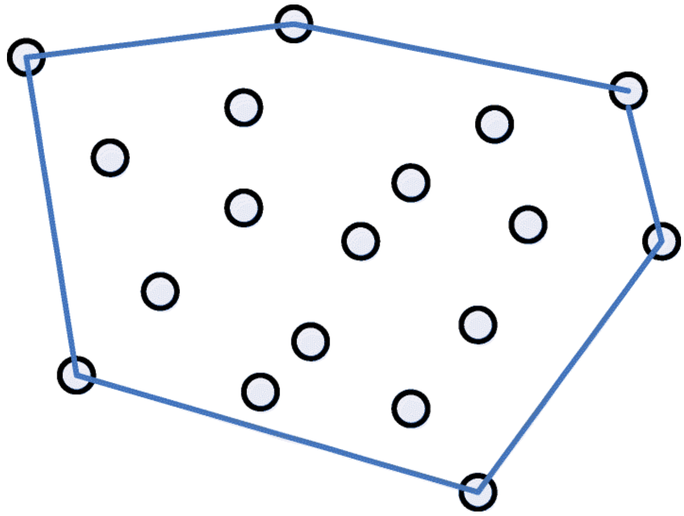
Nyní chci rozebrat novodobý fenomén využití v počítačových technologiích [13]. Budou ukázána dvě použití, a to cloud computing s kyberbezpečností. V prvním případě se jedná hlavně o bezpečnost cloudu z důvodu jeho rozšíření do multi-dimenziálního prostoru. Jedním ze způsobů zabezpečení jsou bezpečné virtuální přístroje využívající Nashovu rovnováhu pro analýzy ve veřejném cloudu [16]. Druhou možností jsou modely sledování risku škálovatelnosti využívaných v reakci na DDOS a podobné útoku s cílem zcizení dat [17]. Nashova rovnováha má své užití v cloudu kromě bezpečnosti též u stanovování cen. Zde se jedná o výpočet ceny takové, která bude výhodná pro obě strany a zároveň poskytovatel získal, co největší možný zisk [18].

Ukázali jsme si, že prvky kyberbezpečnosti jsou částečně propojené s cloudem. Jejím hlavním zaměřením v teorii her je zaměření se na řešení útoků jako SQL injekce, nebo takzvané odmítnutí služby [19]. V daných problémech existují dva hráči: vlastník produktu a zločinec. Jedná se například o ekonomickou stránku, kde je potřeba vzít v úvahu cenu zabezpečení [16]. Nehledě na tuto stránku mají hráči nějaká očekávání o průběhu útoku bez základního historického přehledu, proto postupně adoptují Markovovo rozhodovací řetězce [18].

Celkově lze říct, že zmíněné využití má své limity. V kyberprostoru je těžké kvantifikovat parametry hry [19]. Oproti tomu při počítání her u cloudových poskytovatelů je problém jejich nízký počet, který je zapotřebí pro lepší predikci při větších zatíženích [20].

3.3 Delaunayho triangulace

Při tvorbě triangulace máme k dispozici konečné množství neuspořádaných bodů. Jejich konvexní obal se využívá k jejich uspořádání. Tvorba obalu je spojená s poskytnutím konvexní oblasti bodů pro jeho vznik. Existují různé algoritmy pro nalezení konvexního obalu, my se podíváme na rozděl a panuj, který je časově optimální pro jeho nalezení [12].



Obrázek 4: Konvexní obal bodů v rovině [35]

Konvexnost oblasti platí, pokud jsou dva body v něm sobě vzájemně viditelné. Oblast bodů bude označována S, a intuitivně lze vizualizovat její konvexní obal. Představme si špendlíky zabodané na nástěnce. Obal těchto bodů je vyhrazen pomocí gumičky napnuté okolo všech bodů. Je potřeba pamatovat na pravidlo vzájemné viditelnosti. Jakmile neplatí pro všechny dvojice bodů, jedná se o nekonvexní region. Před uvedením formální definice je nutno zmínit, že si lze díky příkladu s gumičkou představit konvexní obal jako ten nejmenší mezi obaly zahrnujícími všechny body regionu [12].

Definice konvexního obalu. *Konvexní obal S, označen pomocí $conv(S)$, je průnik všech konvexních oblastí obsahujících S [12].*

Nyní se podíváme na princip samotné triangulace neboli rozdělování setu bodů do trojúhelníků. Obecně zde uvádíme skupiny bodů, které nejsou nijak ohraničené. V experimentu této práce budou všechny body omezeny na území jednotkového prostoru. Předtím než se přesuneme na nejdůležitější, a v experimentu využívanou, Delaunayho triangulaci, si ukážeme základní algoritmus a kombinatoriku společně se strukturou zahrnutou v překlápěcím grafu [12].

V našem omezeném prostoru půjde o triangulaci bodů. Vzniklý tvar vypadá obdobně jako obří mnohostěn. Je potřeba rozlišit krajní a vnitřní hrany těchto útvarů. Hrana bude v našem případě označovat spojení právě dvou bodů z S [12].

Definice triangulace. *Triangulace rovinné skupiny bodů S je poddivize roviny určené pomocí maximální setu nekřížících se hran jejichž set vrcholů je S [12].*

Pojem maximální indikuje nutnost křížení vnitřku alespoň jedné hrany triangulace pro jakoukoli hranu, která není v triangulaci [12].

Po uvedení definice je možné uvést algoritmus trojúhelníkové-štěpení pro tvorbu triangulace. Pro jednoduchost předkládejme neexistenci třech kolineárních bodů neboli takových, které neleží na jedné přímce [12].

Algoritmus trojúhelníkové-štěpení. *Nalezněte konvexní obal pro S a triangulujte ho jako mnohostěn. Vyberte vnitřní bod a nakreslete hrany ke třem vrcholům trojúhelníku, které ho obsahují. Pokračujte tento proces do vyčerpání všech vnitřních bodů [12].*

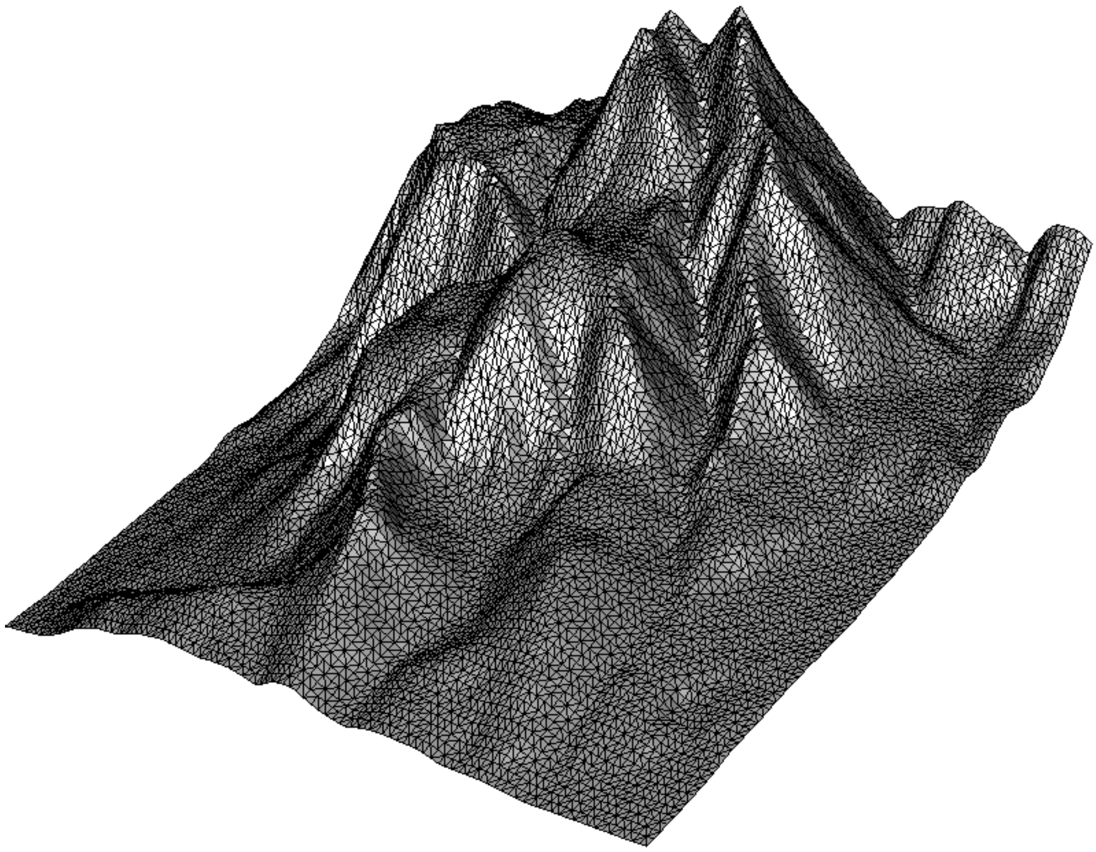
Triangulace skupiny bodů mohou vzniknout různé. Pro všechny takové platí existence stejného počtu trojúhelníků. Jejich počet je určitelný pomocí vzorce. Pokračujeme stále s označením skupiny bodů jako S, pokud označíme vnitřní body k a body tvořící obal jako h dostáváme $2k + h - 2$ určující počet trojúhelníků vytvořených jakoukoli triangulací setu bodů [12].

Překlápěcí graf je jednou z metod zkoumajících strukturu setu bodů S. Zaměřuje se na analýzu triangulací bodů z S. Již bylo uvedeno, že stejný set bodů může mít různé triangulace. Tyto triangulace se mezi sebou mohou lišit pouze o jednu diagonálu, ale také mohou být od sebe velmi odlišné. Tato myšlenka se dá precizně definovat. Nejdříve si představme konvexní obdélník $ABCD$. Dále řekneme, že základní triangulace je tvořená trojúhelníky ABC a ACD. Pojem převrácení okraje změní diagonálu obdélníku tvořící triangulace z AC na BD a vzniknou trojúhelníky ABD s BCD. Takové prohození je možné pouze v konvexních útvarech [12].

Definice překlápěcí graf. *Pro set bodů S je překlápěcí graf S graf, jehož uzly jsou sety triangulací S. Dva uzly T_1 a T_2 z překlápěcího grafu jsou spojeny pomocí oblouku, jestli jedna diagonála T_1 může být překlopena k získání T_2 [11].*

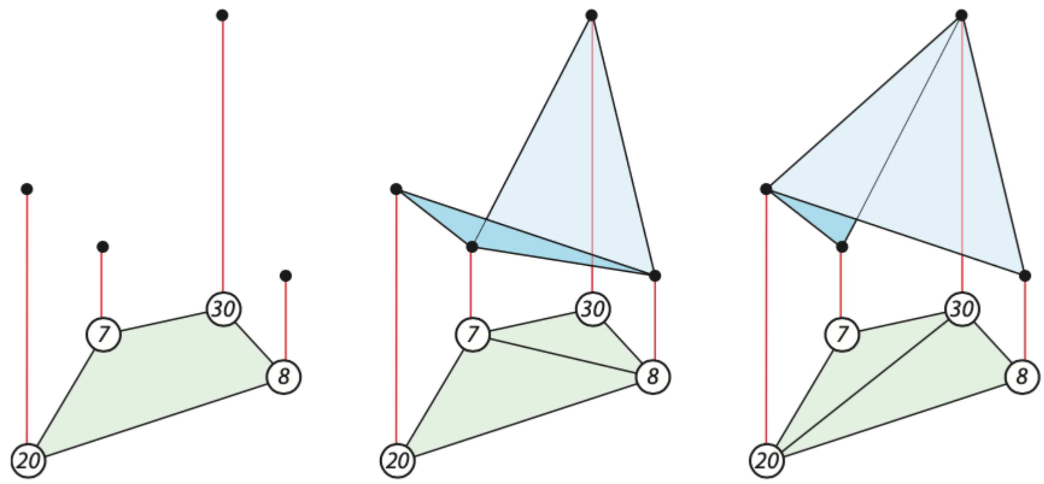
Předchozí definice brali v potaz pouze body v rovině. V této práci simulace vytváří právě takové triangulace. Po jejich vytvoření přiřadí užitky jeho vrcholům, které budou reprezentovat třetí dimenzi [11].

Triangulace se mohou zdát zapouzdřené v překlápěcích grafech, přesto jsou v mnoha případech oceněny některé triangulace více než jiné. Toto tvrzení nás dostává k samotné Delaunayho triangulaci. Nejčastěji může být tento algoritmus pozorován v oblastech rekonstrukce terénu, kterým se práce přibližuje. Daný postup je použit i u tvorby třídimenzionálních map Země. Na počátku tvorby těchto map byl pouze konečný vzorek bodů povrchu S společně s naměřenými výškami. Díky určeným výškám lze určit výšku libovolného blízkého bodu, který nebyl součástí vzorku bodů. Následující proces je přesnou kopií již zmíněného procesu experimentu. Prvním krokem je vytvoření triangulace bez ohledu na výšky bodů. Následně jsou body posunuty do svých správných výšek. Zmíněné dva kroky společně vytvořili trojrozměrné trojúhelníky, které tvoří po částech lineární terén Země [11].



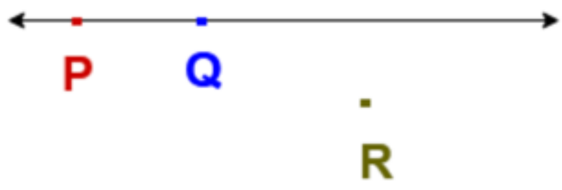
Obrázek 5: Triangulace terénu [38], upraveno

Je třeba si připomenout zmíněný problém různých triangulací vytvořených z jednoho setu bodů S. Není předem jasné, která z nich bude nejvhodnější pro reprezentaci daných bodů. Rovněž tak v realitě není, kromě vzorových bodů, známá reálná podoba terénu. Různé triangulace vytvářejí různé tvary terénu. Některé tyto změny mohou být velmi markantní. Výsledná mapa terénu bude výrazně odlišná při překlopení klíčových hran. V určitých případech je možno z údolí vytvořit pahorek, jako je ukázáno v následujícím obrázku [11].

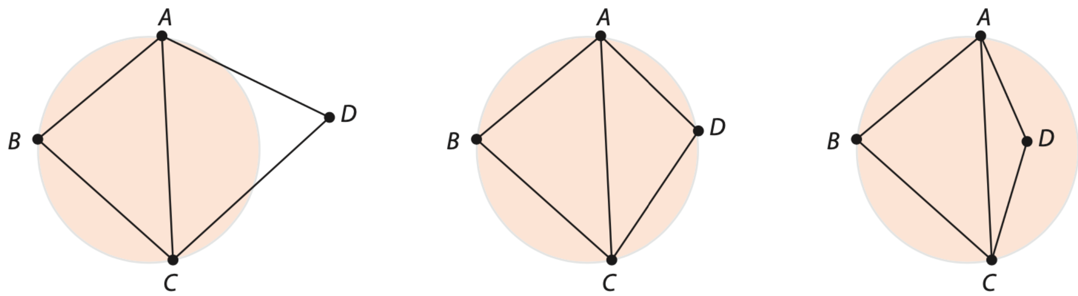


Obrázek 6: Ukázka možných triangulací čtyřúhelníku [11]

Kritérium toho, které triangulace jsou vhodnější můžeme zakládat na své zkušenosti se vzhledem terénu [11]. Díky nim můžeme vybrat z dostupných možností více přirozeně vypadající triangulace. Můžeme tuto intuici aplikovat na obrázku výše. Pojďme označit body čtyrúhelníku po řadě $ABCD$ s výškami $20, 8, 30, 7$ a triangulace T_1 a T_2. V triangulace T_1 vidíme údolí s hranami BD, které jsou přibližně stejně vysoké a vzdálené od sebe menší vzdálenost než úsečka AC. Podle mého subjektivního názoru se jedná o typičtěji pozorovaný fenomén oproti T_2. Ve triangulaci T_2 lze vidět velmi úzký pahorek, který je typický pro extrémní vrchoviny. Daná situace není tolik typickou, alespoň při zkoumání povrhu České republiky oproti T_1, které reprezentuje údolí často pozorovatelné u nás.



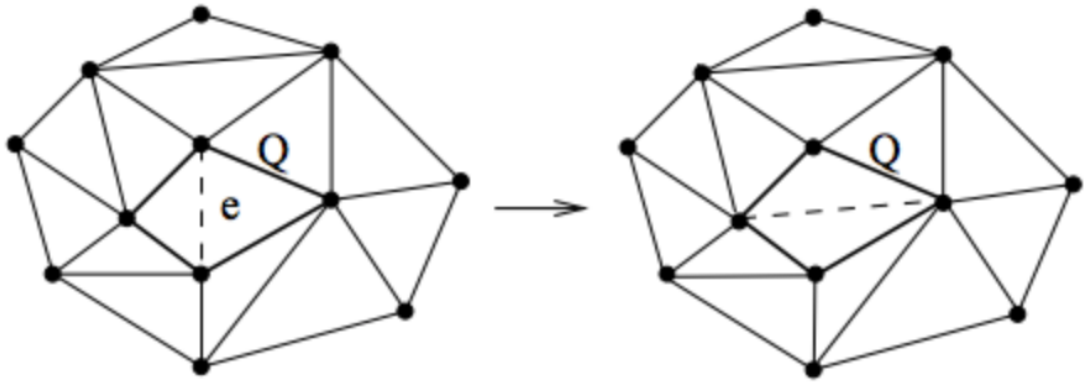
Obrázek 7: Tři nekolineární body [36]



Obrázek 8: Poloha čtyř bodů vzhledem ke kruhu (prostřední případy jsou nechtěné) [11]

Dále uvedeme případ, kdy čtyři body leží na stejné kružnici. Dříve jsme uvedli podmínku neexistence třech kolineárních bodů. Tento fakt označovaný jako obecná pozice bude nyní odpovídat neexistenci kocirkularity. V našem experimentu tento fakt může zřídka nastat. Pojďme označit sekvenci úhlů α v naší triangulaci T vytvořené z bodů S. Bereme v potaz existenci n trojúhelníků, poté získáváme $\alpha = (\alpha_1, \alpha_2, .. \alpha_{3n})$, vzhledem k počtu úhlů v n trojúhelnících. Tyto úhly jsou seřazené podle velikosti. Nyní je možné porovnat dvě triangulace pomocí dané sekvence úhlů. Připomeňme, že všechny triangulace mají stejný počet trojúhelníků [11].

Jedna triangulace je označována tlustší než druhá, pokud má větší lexikografické uspořádání podle sekvence úhlů α. Pro příklad máme dvě následující triangulace, kde první je $T_1 = (10°, 30°, 45°, 65°, 70°, 140°)$ a druhá je $T_2 = (10°, 30°, 45°, 60°, 75°, 140°)$. Při pohledu na T_1 a T_2 vidíme, že na první rozdílné pozici je úhel $65°$, který je větší než $60°$. Z toho vyplývá, že T_1 je tlustší než T_2. Jelikož hledáme co nejtlustší možnou triangulaci, volili bychom mezi těmito dvěma T_1. Chtěnou triangulaci lze získat za pomocí překlápění hran [11].



Obrázek 9: Překlopení hrany e v Q [37]

Definice legální hrany. *Nechť e je hrana triangulace T_1, a nechť Q je čtyřúhelník v T_1 vytvořený dvěma trojúhelníky s e jako společnou hranou. Jestli je Q konvexní, nechť T_2 je triangulace po překlopení hrany e v T_1. Řekněme, že e je legální hrana, pokud $T_1 \geq T_2$ a e je ilegální pokud $T_1 < T_2$ [11].*

Je potřeba si uvědomit rozsah překlápění hran. Po převrácení e jsou změněny všechny úhly α pro triangulace T_1. Ty jsou nahrazeny sekvencí jiných úhlů T_2. Dané překlopení je postaveno pouze na lexikografickém uspořádání. Deklarování všech hran v konvexním obalu jako legální je užitečné pro doplnění definice. Jelikož nelegální hrany nijak nepomáhají v získání nejtlustší triangulace, snažíme se jim vyhnout [11].

Definice Delaunayho triangulace. *Pro set bodů S, je Delaunayho triangulace S, značená $Del(S)$, triangulací s pouze legálními hranami [11].*

Triangulace je pojmenována po ruském matematikovi Borisovi Delaunaym. Není zaručená bezprostřední existence Delaunayho triangulace pro každý bod. Tato triangulace odstraní všechny nelegální hrany bez přidání nových takových hran [11].

Algoritmus Delaunayho triangulace. *Nechť S je set bodů v obecné pozici, kde neexistují čtyři kocirkulární body. Začněme s jakoukoli triangulací T. Jestliže má T ilegální hranu, překlopme hranu na legální. Pokračujme s překlápěním nelegálních hran, postupným procházením překlápěcího grafu S v jakémkoli pořadí, dokud nebudou existovat žádné ilegální hrany [11].*

Sekvence úhlů a se neustále zvětšuje díky definici překlápění hran. Během algoritmu se tak nemůžou opakovat stejné triangulace. Delaunayho triangulace musí skončit, jelikož existuje pouze konečně mnoho uzlů v překlápěcím grafu. Jelikož nezáleží na pořadí překlápění je výsledná triangulace pouze unikátním globálním maximem. Existuje možnost, že napříč všemi uzly existuje i tlustší triangulace [11].

3.4 Domény prostoru

Nyní si blíže probereme tvar domény našeho prostoru. V simulaci fungujeme na rozmezí prostoru $[0,1] \times [0,1]$. Daná doména bude po triangulaci připomínat tvar mnohostěnu.

Mnohostěn představuje mnohoúhelník z dvourozměrného prostoru se zvednutím rozměrů o jednu dimenzi. Tento tvar je reprezentován omezeným regionem tvořeným konečným počtem polygonálních ploch. Před uvedením přesnější specifikace je potřeba začít s konvexními mnohostěny neboli Platonickými útvary. Tento pojem popisuje zobecnění konvexního mnohoúhelníku do třech dimenzí. Dané mnohostěny splňují již zmíněnou konvexitu, tedy propojitelnost jakýchkoli dvou bodů v prostoru bez jeho opuštění. Pro mnohostěny se pak jedná o dihedrální úhly, pro které odpovídá podobná podmínka jako v mnohoúhelnících. Jelikož se v aplikační simulaci nejedná nutně o konvexní útvar je potřeba zvolnit definici regularity, díky tomu budou plochy mnohostěnu moci být reprezentovány několika různými polygony. Při ještě větším rozvolnění podmínky pro povolení nekonvexnosti se dostáváme k požadovaným uniformním mnohostěnům [11]. Právě tento útvar v neuzavřené podobě bude reprezentovat prostor simulované domény.

4 Konkurenční řešení

V této kapitole jsou popsána podobná zpracování tématu. Chci zaměřit na projekty s triangulací prostoru, jelikož se na něm zakládá myšlenka celé práce a při zkoumání podobných výpočtů pro hry s nulovým součtem jsem narazil na shodné zpracování pomocí metody minmax pouze u her s malým strategickým prostorem. Při hledání podobných prací jsem narazil na velké množství takových, které v některé své části využívali metody triangulace. Z nalezených jsem vybral dvě bakalářské a diplomové práce z Českého vysokého učení technického spolu se dvěma vědeckými papíry od profesora Michaela S. Floatera [23].

4.1 Triangulace planárních objektů a jeho implementace do AtoM balíčků

První práce se snaží vybrat nejefektivnější jádro pro generování sítě v AToM, což je akronym pro Antenna Toolbox for MATLAB neboli anténní sada nástrojů pro programovací jazyk MATLAB. Byla vytvořena díky spolupráci českých technických škol mezi lety 2014 a 2017 [24]. Vybraný způsob generování měl poté být implementován do zmíněné sady. Práce byla vybrána, jelikož rozebírá různé možnosti implementace triangulace prostoru. V práci jsou též detailně popsány její prvky spolu s Delaunayho algoritmem [25].

Je potřeba uvést, které algoritmy autor porovnával. Jedná se o DistMesh, Bowyer-Watson, Mesh2D, spolu s triangulacemi jazyka MATLAB jako třída `triangulation` a `delaunayTriangulation`. Překvapivé je, že autor nebere v úvahu třídu `delaunay`, kterou považuji ve své práci jako dostupnou v jazyce MATLAB ke zpracování této problematiky. Ze zmíněných algoritmů se práce rozhodla použít Mesh2D vhledem ke své rychlosti. Kromě algoritmu jsou porovnávány vestavěné vizualizace `triplot`, `trimesh`, `trisurf` [25].

Tato práce má společnou pouze triangulace, je přesto dobré si uvědomit porovnání jednotlivých metod. Oproti autorově práci má moje práce lepší strukturu popisu Delaunayho algoritmu, jelikož ve zmíněné práci jsou nadefinovány prvky, které nejsou potřebné k pochopení problematiky. Po porovnání s touto prací budu uvažovat, zda nemohou být některé z porovnávaných algoritmů využity v mé práci. Též mi srovnání pomohlo si uvědomit důležitost vizualizace jednotlivých problematik během jejich teoretického popisu.

4.2 Julia rozhraní pro knihovnu Triangle

Oproti první srovnávané bakalářské práci se druhá snaží navrhnout a implementovat nové rozhraní pro knihovnu Triangle v jazyce Julia. Teorie v této práci vysvětluje stejné teoretické prvky potřebné pro porozumění Delaunayho algoritmu triangulace jako v mé práci. Navíc je zde popsán Voroného diagram. Opět jsem si potvrdil důležitost vizualizace, a přidal jsem díky této práci do klíčových míst popisu algoritmu obrázky pro lepší porozumění problematiky [26].

Jak bylo zmíněno, práce pracuje s jazykem Julia a má za cíl implementovat knihovnu Triangle, psanou v C. Tento jazyk je spolu s Julia je rychlejší než MATLAB [27], avšak MATLAB poskytuje interaktivní prostředí, kompilaci při spuštění a několik knihoven pro matematické operace [28], které ve své odvozené interpretaci byly méně přesné v mých předchozích experimentálních simulacích prováděných v jazyce Python. Ten je napsán pomocí jazyka C a v mé práci je potřeba co nejpřesnějších výsledků vzhledem k potenciálně nekonečnému zjemňování.

Přestože se opětovně jedná pouze o část mé práce je zde vidět několik odlišností. Porovnávaná práce řeší pouze problém triangulace, tedy pro svou implementaci potřebuje více tříd, které reprezentují všechny autorem definované pojmy. S tím se pojí nutnost více názvů proměnných, které jsou potřeba popsat a v autorově práci mi nepřijde pár slov dostatečných k pochopení. Oproti tomu se v mé práci mohu prvkům triangulace více věnovat, a tedy lépe proměnné algoritmu popsat k jejímu lepšímu porozumění. Zároveň implementuje několik typů Delaunayho triangulací, které v mé práci nejsou potřeba vzhledem k využití této triangulace pro vytvoření trojúhelníků pouze v prvním kroku simulace pro malé množství bodů.

4.3 Výpočet objemu objektů z rozsáhlých zašuměných mračen bodů

Přesouváme se k diplomovým pracím. První z nich opět využívá tvorbu triangulačních sítí jako předchozí porovnávaná práce. Stejně tak slibuje detailní popsání již opakovaného algoritmu Delaunayho triangulace. Oproti předchozím je autorčina práce implementována ve prostředí shodném s mým, tedy v jazyce MATLAB [29].

V porovnání prací mi nepřijde popis práce vůbec detailní. Přes svou délku používá k popisu výpočet determinantu pro ověření kocirkularity, který nebyl v ostatních pracích uveden. Mimo to jsou pouze uvedeny vlastnosti triangulace v bodech. Při srovnání teoretických částí je v mé práci podrobněji popsán tento algoritmus pro lepší pochopení. Pokud by byla obhajována předchozí znalost vzhledem ke zpracování diplomové práce, pak není potřeba uvádět rok vzniku algoritmu [29].

Porovnávaná práce se zabývala speciálním případem 2.5D prostoru s neorganizovanými mraky bodů. Nejprve je tato metodika použita pro výpočet objemu hlučných bodů připomínajících známé útvary krychle a koule pomocí triangulace, poté pro reprezentaci obrázku Karlova náměstí. Při spuštění trval běh algoritmu triangulace tohoto náměstí na domácím notebooku 14 hodin a vytvořil šest set tisíc trojúhelníků [29].

Tyto dvě práce se opět shodují pouze v použití triangulace. Popis Delaunayho algoritmu je lépe popsán v této oproti autorčině práci. Porovnávaná práce má oproti předchozím dvěma navíc výpočet determinantu pro ověření kocirkularity. Při porovnávání jsem se opět přesvědčil o důležitosti vizualizace jednotlivých kroků nejen v teoretické části, což ovlivnilo její frekvenci v mé práci.

4.4 Viditelnost triangulovaného povrchu osvětleného plošným osvětlovačem

Další práce se zabývá něčím, co bylo vidět ve velkém množství prací pracujících s triangulací. Jedná se o senzorické snímání, které obraz přetváří na triangulace.

V tomto případě se jedná o použití kamery. Autorův algoritmus rozpoznává viditelné části povrchu a je schopný zpřesnit triangulaci [30].

Podíváme se opět na teoretickou část triangulace. V porovnávané práci začíná tato část krátkým úvodem následovaným definicí trojúhelníku, která mi přijde nevhodná do práce této úrovně. Na úkor tomu jsou následující kapitoly věnující se triangulaci a Delaunayho triangulaci krátké podobně jako v předchozí diplomové práci. Kapitola triangulace začíná popsáním domény zahrnující všechny body a pokračuje podmínkou o kolinearitě bez definice. Při popisování Delaunayho triangulace není podmínka nejtlustší triangulace vysvětlena, namísto toho je zmíněná nutnost co největšího nejmenšího úhlu. Po zmínění úhlů je definována podmínka kružnice neboli kocirkularity [30].

Opět se nejedná o hlavní část práce, přesto je Delaunayho triangulace jejím klíčovým algoritmem. Podle mého názoru se opět jedná o nedostatečně vysvětlenou problematiku oproti mé práci. Oproti tomu je dobře vše vizuálně ukázáno, což potvrzuje podstatu obrázků v teoretické části.

4.5 Algoritmy filtrů banky pro po částech lineárních předvlnek na libovolných triangulací

Přesouváme se do části výzkumných prací pod vedením pana Michaela S. Floatera. V první z nich jsou zkoumány dekompoziční, rekonstrukční a aproximační algoritmy na po částech lineárních předvlnkách na ohraničených triangulacích. Autoři se snaží ukázat symetrii, pozitivní definitnost a dobrou podmíněnost Schurova doplňku. Pomocí numerických příkladů ukazují autoři menší aproximační chybu oproti základní Faberovu dekompozičnímu schématu [31].

V jedné části porovnávané práce jsou tyto metody porovnávány na dvou příkladech aproximace terénu pomocí výpočtů matematické analýzy nad různě jemnými triangulacemi. Proces zjemnění zde není nijak popsán. Na některých obrázcích se jedná o provedení jemnější triangulace za použití již vytvořené základní, avšak na ostatních se jedná o jinou triangulaci spolu s jemnějším základním nastavením. Přestože tato práce nebude triangulaci zjemňovat, bude vytvářeno zjemnění prostoru pomocí tvorby průsečíků s jejími úsečkami. Kdyby těmito novými body byly vedeny nové úsečky dostaneme se pro jednotlivé trojúhelníky podobnému tvaru [31].

Výpočty srovnávané práce používají integrální počet pro srovnání chyby odlišných výpočtů [31]. Oproti tomu se moje práce spoléhá na duální lineární program pro svou přesnost. Naopak se práce shodují v použití jazyka MATLAB pro matematickou vizualizaci. V porovnávané práci není zmínka o použitém jazyku, přesto na to poukazuje vzhled obrázků.

4.6 Po částech lineární vlnky přes triangulace druhého typu

Další výzkumný článek profesora Floatera byl vybrán kvůli využívání zjemněných triangulací spolu s lineárními funkcemi uprostřed reprezentace domény pozorovaného prostoru. Článek představuje metodu zjemnění pomocí půlení všech existujících úseček a následného dotvoření nových trojúhelníků. Pokud si odmyslíme nové úsečky, můžeme při zkoumání vytvořené množiny vrcholů vidět, že se jedná o nadmnožinu bodů oproti mé množině, která by vznikla, kdybychom na triangulaci spustili náš program pro zjemnění mřížky. V algoritmu článku se rozpůlí každá úsečka novým vrcholem. Oproti tomu se v mé práci hledají pouze horizontální a vertikální průsečíky úseček triangulace vedených z původních vrcholů [32].

Druhou podobností je reprezentování vrcholů nějakými hodnotami. Ze článku nelze rozpoznat, zda se jedná o výšky a jak jsou hodnoty kalkulovány. Jelikož se v mé práci jedná o hodnoty jednotlivých užitkových hodnot strategií, jsou počítány pomocí různých metod analytické geometrie pro zajištění jejich přesnosti. Mé hodnoty se dají představit tak, že při triangulaci kopce jsem schopen pro každý bod určit jeho výšku. U článku nelze jasně říct, jaké jsou hodnoty původních vrcholů, jelikož to není ve článku vizuálně zpracováno a ukázáno. Přesto se dá tvrdit, že se nejedná o výpočet velikosti bodů [32].

5 Návrh algoritmu

Po předchozím popisu teorie se můžeme přesunout k naprogramování řešení. Algoritmus nejdříve náhodně vybere body na vybrané doméně $[0,1] \times [0,1]$. Tyto body následně pomocí Delaunayho algoritmu implementovaným vestavěnou funkcí `delaunay`. Výstupem této funkce je list trojic indexů náhodných bodů tvořících trojúhelníky. Jelikož není tento tvar vhodný pro všechny potřeby programu, je též přidána verze se souřadnicemi bodů a také list jednotlivých úseček. Ten je využit hlavně pro jednodušší hledání průsečíků, při procesu tvorby a zjemňování požadované mřížky na doméně.

Práce nad mřížkou je prováděna iterativním postupem. V něm se nejdříve naleznou pro všechny aktuálně dostupné body průsečíky s úsečkami triangulace a je vypočten jejich užitek. Po takovémto projití všech bodů je ze všech vytvořen kartézský součin pro doplnění mřížky reprezentující strategický prostor nulové hry. Body vytvořené až díky kartézskému součinu potřebují dopočítat užitek. To je zajištěno pomocí nalezení trojúhelníku, do kterého náleží a následného výpočtu.

Po dopočítaní posledního chybějícího užitku je možno spočítat herní užitek obou hráčů v naší hře s nulovým součtem. Nejdříve je potřeba vzít užitky z listu všech bodů a vytvořit pomocí nich matici užitků. K vyhodnocení hry se využívá funkce `linprog`, která postupuje podle popisovaného algoritmu ve druhé kapitole. Výsledek této funkce vrací pravděpodobnostní rozdělení napříč strategiemi spolu s celkovým hráčským užitkem.

Posledním krokem je grafické znázornění pravděpodobnostního rozdělení pomocí funkce `stairs`. Obdobně jsou znázorněny všechny aktuální body na doméně pomocí `triplot`, `plot`, a `plot3` u dvoudimenzionálního zobrazení a `trimesh` při třídimenzionálního.

Následně se algoritmus vrací do startovního bodu iterace díky čemuž je mřížka zjemňována a více bodů generováno. Tato iterace je opakována podle uživatelem specifikované hodnoty. Program ukazuje, že se strategické pravděpodobnostní rozdělení nemění při jemnější mřížce.

6 Implementace

Program si na začátku svého běhu postupně zeptá, zda chce uživatel nastavit vstup. Ten se skládá ze čtyř proměnných. Jmenovitě se jedná o počet generovaných bodů, jemnotu mřížky na doméně, počet běhů zjemňování, a rozmezí užitkových hodnot neboli výšek. Uživatel má možnost využít přednastavených hodnot.

OBRAZEK CHECKVALIDITY

Pokud uživatel zadává hodnoty sám, tak jsou následně kontrolované, zda vyhovují požadavkům pomocí funkce checkValidity. U bodů je potřeba zvolit číslo větší než 4, jelikož jsou na doméně vždy voleny krajní body. Minimální jemnost je nastavena na 0.1. Tato hodnota umožňuje sto míst k náhodnému umístění specifikovaného počtu bodů. Třetím parametrem je počet běhů iteračního zjemňování mřížky bodů reprezentujících užitky. Zde je potřeba alespoň hodnota 1, aby mohl algoritmus doběhnout do konce a spočítat hráčova pravděpodobnostní rozdělení mezi strategiemi. Samotný jeden běh ovšem nedokáže ukázat nezávislost jemnosti bodů na pravděpodobnostním rozdělení. Běhy algoritmu bez procesu zjemňování slouží hlavně k zobrazení prostoru bodů a vyhodnocení základní nulové hry. Poslední parametr určuje maximální hodnotu užitku pro body. Tento užitek je na doméně zároveň jejich výškou. Body dostanou přidělenou náhodnou hodnotu od 0 až do specifikované hodnoty.

OBRAZEK GENEROVANI PROSTREDI

Po zpracování parametrů přichází na řadu generování základního vzhledu domény, který bude ovlivněn polohou náhodného rozmístění zvoleného počtu bodů. Zmíněná generace je zpracována funkcí `generateEnvironment`, která si jako argumenty bere, kromě počtu běhů, všechny vstupy. Nejdříve se pomocí jemnosti zjistí, kolik je dostupných míst pro body. Díky tomu je možné vytvořit dvoudimenzionální matici `pointsArr` těchto rozměrů s hodnotami -1 na všech pozicích a přidat body na rohové indexy domény, poté lze náhodně vybrat zbytek bodů mezi zbylými indexy. Následně je pro všechny body přiřazena náhodná výška.

V matici pointsArr jsou vybrané indexy bodů nahrazeny hodnoty výškami těchto bodů a funkce ji poté vrátí.

K tomu, aby se lépe četlo umístění bodů jsou data z pointsArr převedena na souřadnice bodů. Proměnná coords bude tyto údaje uchovávat. Nové body budou přibývat, proto je vytvořená kopie coordsForEqs, která je předávána do funkcí, kde jsou potřebné pouze původní body.

OBRAZEK DELAUNAY

Nutnost souřadnic je viditelná již u další funkce, která se stará o vytvoření triangulace. Jedná se o vestavěnou funkci delaunay, která si jako argumenty bere listy souřadnic x, y a vrací list trojúhelníků triangArr specifikující jejich indexy v těchto listech. Jak již bylo zmíněno v návrhu funkce postupuje podle Delaunay algoritmu popsaného v teoretické části. Z listu triangArr se dále získává list úseček lineList pomocí projití všech kombinací ve vytvořených trojúhelnících, a poté list úseček s koordináty lLwCoords díky procházení těchto úseček a přiřazování úseček do něj. Tyto dvě hodnoty jsou získány voláním funkce createListOfLines s argumenty coords a triangArr. Proměnná lLwCoords se následně seřadí do sLwCoords.

OBRAZEK TRIANGLE EQUATION

Pro budoucí potřeby je nutnost popsat jednotlivé trojúhelníky, aby šly užitky jejich bodů správně vypočítat. To zpracovává funkce calculateTriangEq, která si bere jako parametry triangArr a coordsForEqs. Funkce předpočítá potřebné hodnoty pro výpočet determinantu u hledání vhodného trojúhelníku. Návratovými hodnotami jsou cTriangleVariables a triangleEquations.

6.1 Iterace

Iterace zajišťuje v programu nalezení průsečíků pro body a následné vytvoření kartézského součinu pro zhotovení mřížky. Jakmile jsou přidány všechny užitky bodům, se iterace posouvá do fáze vyhodnocení hry s nulovým užitkem našich dvou hráčů. Následně jsou vyobrazeny body na doméně spolu s pravděpodobnostním rozdělení napříč strategiemi hráčů.

OBRAZEK SMOOTHENPLANE

Nyní postupně ukážeme, jak jsou kroky implementovány. Funkce `smoothenPlane` bere argumenty `coords` a `sLwCoords`. Seřazený list se využívá k tomu, aby šlo lépe sledovat hledání průsečíku. Do proměnné `smootherpoints` se zkopírují body z `coords`, kam budou přidávány nové body. Funkce prochází všechny body z `coords`. Pro zrychlení procesu je na místě pár vylepšení. Jelikož chceme najít všechny průsečíky v horizontálním a vertikálním směru, je jedním z nich vybrání pouze úseček, které mají potenciál tvorby průsečíku. Tento proces je prováděn pro vertikální i horizontální hledání. Pro oba směry je vypočtení průsečíku rozděleno na úsečky kolmé a ty ostatní. To je dáno tím, že pro kolmé úsečky budou mít nové body hodnotu jedné ze souřadnic rovnou hodnotě úsečky v závislosti na směru. Druhou vychytávkou je omezení duplikátů. To je provedeno tak, že když se jedná o bod se stejnou souřadnicí ve směru jako předtím kontrolovaná, pak ho současný bod přeskakuje. Po nalezení nového bodů je vypočítán jeho užitek. U nových průsečíků se volá funkce `calculateHeights` s parametry jeho a celé úsečky, kterou protnul. Tato funkce funguje na principu analytické geometrické, kde se dosadí souřadnice nového bodu do úsečky a poměrově se přičte absolutní hodnota rozdílu užitku krajních bodů podle vzdálenosti od kraje s menším užitkem. Jakmile funkce projde všechny body a nalezne možné průsečíky je potřeba ověřit, zda se nenachází nějaký duplikát. Body jsou zaokrouhleny na čtyři desetinná místa a pomocí vestavěné funkce `unique` se takové duplikáty odstraní. Následně množinu bodů `smootherPoints` vrátí.

OBRAZEK CARTESIAN

K dokončení mřížky je potřeba vytvořit kartézský součin všech dostupných bodů. Tento výpočet zastává funkce `makeCartesianProduct`, která bere jako jediný argument `smootherPoints`. Uvnitř funkce se vyberou unikátní x-ové a y-ové souřadnice a pomocí vestavěné funkce `combvec` utvoří všechny jejich kombinace. Tím vznikne proměnná `cartesianPoints`, která současně obsahuje pouze první dva koordináty. Jelikož bude tato proměnná obsahovat všechny užitky, tak dostává třetí rozměr plný hodnot -1. Pomocí vestavěné funkce `intersect` se naleznou indexy již spočítaných užitků, které jsou uloženy v `smootherPoints`. Tyto hodnoty se přiřadí na dané indexy proměnné `cartesianPoints`. Pro zbylé body existuje funkce `calculateCPHeights`, která má několik argumentů. Postupně se jedná o `triangArr`, `coordsForEqs`, `cartesianPoints`, `triangleEquations`, a `cTriangleVariables`. Uvnitř funkce se procházejí pouze body bez užitku, pro které se prochází trojúhelníky s cílem nalezení takového, kterému náleží. Vzorec je rozepsaný determinant jednotlivých vrcholů trojúhelníku s bodem. Pokud jsou všechny takové hodnoty kladné, nebo záporné, pak se jedná o bod náležící trojúhelníku. Jeho užitek je dopočítán pomocí polohy na ploše trojúhelníku.

Nyní se dostáváme do fáze vyhodnocení hry. To má na starost funkce `evaluateZeroSumGame`, která si bere jako argument proměnnou `cartesianPoints`. Tato proměnná obsahuje list a duální lineární program vyžaduje matici. K jejímu získání se využívá vestavěná funkce `groupcounts`, která vrací kvantitu jednotlivých souřadnic. K získání rozměrů je vzat první prvek jako jeden rozměr a druhý je dopočítám vydělením velikost proměnné `cartesianPoints` prvním získaným rozměrem. Vestavěná funkce `linprog` je používaná k výpočtu strategického profilu a celkového užitku obou hráčů pro danou matici užitků. Pro náš duální problém se jedná o několik proměnných, které jsou potřeba vytvořit, abychom je mohli předat funkci ke zpracování. Ukážeme si tyto proměnné pro jednoho hráče, který se jmenuje Alice. Ta se bude snažit minimalizovat hodnotu řádkového užitku.

OBRAZEK LINPROG

Mezi požadované parametry patří funkce f, matice A, pravá strana b, součet hodnot neznámých A_{eq}, hodnota tohoto součtu b_{eq}, spodní omezení lb, a horní omezení ub. Pro vizualizaci je vhodné upravit popsaný duální program.

$$\min x_0$$

$$za\ podmínek\ A * x_j - 1 * x_0 \geq 0$$

$$\sum_{j \in A_1} x_j = 1$$

$$x_j \geq 0 \quad \forall j \in A_1$$

Odshora postupně dostáváme f, které požaduje hodnotu všech proměnných, tedy pro všechna x_j je hodnota nula a pro x_0 je hodnota 1. Též je potřeba mít všechny x-ové proměnné na jedné straně, tedy přilepíme x_0 k matici A a pojmenujeme ji A_m. Pravá strana se rovná nulovému vektoru o velikosti A_2. Opačně od funkce f je potřeba uvnitř parametru A_{eq} označit všechny x_j hodnotou 1 a x_0 nulou. Proměnná b_{eq} se rovná 1 ze třetího řádku. Ohledně omezení máme pouze spodní omezení lb rovno nule pro všechny x_j, ub se rovná prázdné množině. Obdobně je to u druhého hráče pojmenovaného Bob. Oproti Alici se bude jednat o maximalizaci sloupcového užitku. Je tedy potřeba transponovat matici a využívat opačný rozměr. Též je potřeba přidat mínus před funkci f a matici, která je tentokrát pojmenována A_n pro odlišení hráčů. Návratová hodnota funkce vrací pravděpodobnostní rozdělení společně s užitkem. Z toho důvodu je od užitku odděleno pro pozdější vizualizaci. Při kontrole výsledků jsou podmínky lineárního programu splněné a přesouváme se na vizualizace.

OBRAZEK NAHODNE TRIANGULACE 2D, 3D, UZITKU (popsat souradnicove osy)

Ke znázornění bodů dvourozměrně je využit `triplot` a pro třírozměrně `trimesh`. Toto zobrazení je zpracováno funkcí `displayTriangulated`, která si bere `triangArr`, `heightList`, x a y koordináty, `cartesianPoints`, a pravděpodobnostní rozdělení jako argumenty. Vizualizovaní strategického profilu je obstaráno vestavěnou funkcí `stairs` vzhledem k její vhodnosti pro vyobrazení hustoty pravděpodobností mezi všemi strategiemi. Po tomto vyobrazení je iterace u konce a v případě, že je hodnota `numRuns` větší než jedna, je dále opakován, dokud nejsou všechny iterace vypočtené.

7 Vyhodnocení výsledků

Klepněte sem a začněte psát text Vaší práce

Výsledky potvrzuji predpoklad.

Jak dlohu se pocitji nove body

Ukázka iteraci 1, 3, 5, 7

8 Závěr

Klepněte sem a začněte psát text závěru Vaší práce

Seznam bibliografických odkazů

[1] HYKŠOVÁ, M.: *Historical Beginnings of Game Theory*. [online]. [citováno 13.10.2021]. Dostupné z: http://euler.fd.cvut.cz/predmety/game_theory/history.pdf

[2] KLIMM, M.: *Games and Their Equilibria*. 2016. [online]. [citováno 15.10.2021]. Dostupné z: https://www.coga.tu-berlin.de/fileadmin/i26/download/AG_DiskAlg/FG_KombOptGraphAlg/klimm/AGT/chapter01.pdf

[3] KUBIŠ, Stanislav: *Hry s po částech afinními užitkovými funkcemi*. 2021. [online]. [citováno 15.10.2021]. Dostupné z: https://dspace.cvut.cz/bitstream/handle/10467/92782/F3-BP-2021-Kubis-Stanislav-Games_with_Piecewise_Affine_Utility_Functions.pdf

[4] LEYTON-BROWN, Kevin; and SHOHAM, Yoav: *Essentials of Game Theory*. [online]. [citováno 25.10.2021]. ISBN: 9781598295948. Dostupné z: http://physics.ujep.cz/~jskvor/KVM/TeorieHer/shoham.pdf

[5] OSBORNE J., Martin; RUBINSTEIN, Ariel: *A Course in Game Theory* (verze roku 2012). Nakladatelství The MIT Press, 1994. [online]. [citováno 21.10.2021]. Dostupné z: https://arielrubinstein.tau.ac.il/books/GT.pdf. ISBN: 0-262-65040-1.

[6] CARSE, James P: *Finite and Infinite Games*. New York: Ballantine Books. 1986. [online]. [citováno 2.11.2021]. ISBN 978-0-345-34184-6. Dostupné z: https://archive.org/details/finiteinfinitega00carsrich/mode/2up

[7] OZDAGLAR, A.: *Lecture 6: Continuous and Discontinuous Games*. 2010. [online]. [citováno 2.11.2021]. Dostupné z: https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-254-game-theory-with-engineering-applications-spring-2010/lecture-notes/MIT6_254S10_lec06b.pdf

[8] SHOHAM, Y. and LEYTON-BROWN, K.: *MULTIAGENT SYSTEMS Algorithmic, Game-Theoretic, and Logical Foundations*. Cambridge: Cambridge University Press . 2009. [online]. [citováno 5.11.2021]. Dostupné z: http://www.masfoundations.org/mas.pdf

[9] SLOUGHTER, D.: *Section 1.5 Linear and Affine Functions*. 2001. [online]. [citováno 7.11.2021]. Dostupné z: http[]://cfsv.synechism.org/c1/sec15.pdf

[10] TADELIS, S.: *Mixed Strategies*. [online]. [citováno 7.11.2021]. Dostupné z: http://faculty.haas.berkeley.edu/stadelis/Game%20Theory/econ160_mixed.pdf

[11] DEVADOSSL.S.; and O'ROURKE J.: *Discrete and Computational Geometry*. Princeton University Press Princeton and Oxford. 2011. [citováno 11.11.2021]. ISBN: 978-0-691-14553-2.

[12] DASKALAKIS, Constantinos; GOLDBERG, Paul W.; and PAPADIMITRIOU, Chistos H.: *The Complexity of Computing a Nash Equilibrium*. 2008. [online]. [citováno 15.11.2021]. Dostupné z: https://people.csail.mit.edu/costis/simplified.pdf

[13] HYKŠOVÁ, Magdalena: *Several Milestones in the History of Game Theory*. [online]. [citováno 15.11.2021]. Dostupné z: http://euler.fd.cvut.cz/~hyksova/hyksova_milestones.pdf

[14] SANDHOLM, William H.: *Lecture Notes on Game Theory and Information Economics*. 2019. [online]. [citováno 16.11.2021]. Dostupné z: https://www.ssc.wisc.edu/~whs/gtie.pdf

[15] BECKER, Robert A.: *Game Theory with Applications to Economics, 2d ed*. 1992. James W. Friedman, The Journal of Economic. Education. [online]. [citováno 16.11.2021]. Dostupné z: https://www.tandfonline.com/doi/abs/10.1080/00220485.1992.10844743?journalCode=vece20

[16] YUANW.; YONGJUN W.; JING L.; ZHIJIAN H.; and PEIDAI X.: *A Survey of Game Theoretic Methods for Cyber Security*. IEEE First International Conference on Data Science in Cyberspace. 2016. [citováno 16.11.2021].

[17] BURCHN.: *Time and Space: Why Imperfect Information Games are Hard.* Ph.D. Dissertation, University of Alberta. 2017. [citováno 16.11.2021].

[18] KAKKAD, V.; SHAH, H.; PATEL R. and DOSHI N.: *A Comparative Study of Applications of Game Theory in Cyber Security and Cloud Computing*. Pandit Deendayal Petroleum University, Gandhinagar, India. 2019. [online]. [citováno 16.11.2021]. Dostupné z: https://www.sciencedirect.com/science/article/pii/S1877050919310130

[19] AMADI CH.; EZE U.; a IKERIONWU CH.: *Game Theory Basics and Its Application in Cyber Security*. Advances in Wireless Communications and Networks. 2017. [citováno 16.11.2021].

[20] SHIVAS.; SANKARDAS, R.; a DIPANKAR, D.: *Game Theory for Cyber Security*. ACM Computing Surveys. 2010. [citováno 16.11.2021].

[21] TSANG, Y-K.: *MATH 1020 Chapter 1: Introduction to Game theory*. 2016. [online]. [citováno 15.11.2021]. Dostupné z: https://slideplayer.com/slide/9860380/

[22] HYKŠOVÁ, M.: *Normal Form Games*. [online]. [citováno 20.11.2021]. Dostupné z: http://euler.fd.cvut.cz/predmety/game_theory/lecture_introduction_normal.pdf

[23] Nettredaksjonen, Matematisk institut: *Michael S. Floater*. 2013. [online]. [citováno 27.11.2021]. Dostupné z: https://www.mn.uio.no/math/english/people/aca/michaelf/

[24] MAŠEK, M.: *ABOUT THE TEAM*. 2021. [online]. [citováno 27.11.2021]. Dostupné z: http://www.antennatoolbox.com/about-us

[25] ŠTRAMBACH, M: *Triangulation of Planar Objects and Its Implementation into AToM Package*. 2017. [online]. [citováno 27.11.2021]. Dostupné z: https://dspace.cvut.cz/bitstream/handle/10467/69334/F8-BP-2017-Strambach-Martin-thesis.pdf?sequence=1&isAllowed=y

[26] KUZMA, M.: *Julia rozhraní pro knihovnu Triangle*. 2017. [online]. [citováno 29.11.2021]. Dostupné z: https://dspace.cvut.cz/bitstream/handle/10467/69160/F8-BP-2017-Kuzma-Martin-thesis.pdf?sequence=1&isAllowed=y

[27] KVASNICKA, M.: *Matlab is significantly slower than Julia on simple evaluation*. 2021. [online]. [citováno 29.11.2021]. Dostupné z: https://www.mathworks.com/matlabcentral/answers/1582569-matlab-is-significantly-slower-than-julia-on-simple-evaluation

[28] FANGOHR, H.: *A Comparison of C, MATLAB, and Python as Teaching Languages in Engineering*. 2004. [online]. [citováno 29.11.2021]. Dostupné z: https://link.springer.com/content/pdf/10.1007%2F978-3-540-25944-2_157.pdf

[29] TŮMOVÁ, Z.: *Object Volume Calculation from Large Noisy Point-Clouds*. 2018. [online]. [citováno 30.11.2021]. Dostupné z: https://dspace.cvut.cz/bitstream/handle/10467/77046/F3-DP-2018-Tumova-Zuzana-Diplomka.pdf?sequence=-1&isAllowed=y

[30] KOUCKÝ, L.: *Visibility of Triangulated Surface Illuminated by Flat Light Panel*. 2014. [online]. [citováno 30.11.2021]. Dostupné z: https://dspace.cvut.cz/bitstream/handle/10467/24607/F3-DP-2014-Koucky-Lukas-prace.pdf?sequence=3&isAllowed=y

[31] FLOATER, M. S.; QUAK, E. G.; a REIMERS, M.: Filter Bank Algorithms for Piecewise Linear Prewavelets on Arbitrary Triangulations. [online]. [citováno 30.11.2021]. Dostupné z: https://www.mn.uio.no/math/english/people/aca/michaelf/papers/filter.pdf

[32] FLOATER, M. S.; a QUAK, E.: Piecewise Linear Wavelets over Type-2 Triangulations. [online]. [citováno 30.11.2021]. Dostupné z: https://www.mn.uio.no/math/english/people/aca/michaelf/papers/type2.pdf

[33] EISERT, J.: *Entanglement in quantum information theory*. 2006. [online]. [citováno 9.12.2021]. Dostupné z: https://www.researchgate.net/figure/The-pay-off-table-in-the-so-called-Matching-Pennies-game-The-first-entry-refers-to_fig15_34931515

[34] MU, Y.: *A New Class of Control Systems Based on Non-equilibrium Games*. 2010. [online]. [citováno 9.12.2021]. Dostupné z: https://www.researchgate.net/figure/The-payoff-matrix-of-Rock-Paper-Scissors-game_fig3_266997381

[35] LIU, R; TANG, Y.; a CHAN, P.: A *fast convex hull algorithm inspired by human visual perception*. 2018. Multimedia Tools and Applications. 2018. [online]. [citováno 9.12.2021]. Dostupné z: https://www.researchgate.net/figure/The-definition-of-convex-hull-In-the-figure-the-polygon-is-the-convex-hull-of-the-point_fig1_325578736

[36] SIVARAMAKRISHNA, A.: *non-collinear-points*. [online]. [citováno 9.12.2021]. Dostupné z: https://www.allmathtricks.com/point-collinear-noncollinear/non-collinear-points/

[37] O'ROURKE, J.: *Do random triangulations edge-flips maintain randomness?* [online]. [citováno 9.12.2021]. Dostupné z: https://mathoverflow.net/questions/209403/do-random-triangulation-edge-flips-maintain-randomness

[38] KORNBERGER, B.: 2.5D Terrain Triangulation and Point Cloud Simplification. 2016. [online]. [citováno 9.12.2021]. Dostupné z: https://www.geom.at/terrain-triangulation/

UKÁZKA:
TANENBAUM, Andrew S. a Albert S. WOODHULL. *Operating systems: design and implementation*. 3rd ed. Upper Saddle River, N. J.: Pearson/Prentice Hall, c2006. ISBN 978-0131429383.

K vytvoření citace doporučujeme použít server www.citace.com

Seznam příloh

Příloha A Klepněte sem a napište název přílohy

Klepněte sem a vložte svou přílohu nebo napište doprovodný text přílohy