1. Úvod

V tejto práci je riešená implementácia modelu (IMS, 7) reštauračného systému (IMS, 7) konkrétnej prevádzky s názvom Iná haluška v Trnave, ktorý simuluje proces (IMS, 124) obsluhy zákazníkov, prípravy jedál a počet zákazníkov počas rôznych častí dňa. Model umožňuje analyzovať efektivitu prevádzky a identifikovať optimálny počet personálu.

1.1 Význam projektu

Simulácie (IMS, 8) boli realizované na základe údajov získaných z reštaurácie Iná haluška. Pomocou simulačných experimentov bude ukázané, či je počet čašníkov a kuchárov adekvátny vzhľadom na rôznu frekvenciu príchodov zákazníkov. Zmyslom experimentov je posúdiť, ako vyťaženosť personálu ovplyvňuje čakacie doby zákazníkov a efektivitu prevádzky, a tým overiť, či aktuálne rozdelenie pracovnej záťaže spĺňa požiadavky na spokojnosť zákazníkov.

1.2 Náročnosť projektu

Pre spracovanie modelu bolo nutné pochopiť princípy diskrétnej (IMS, 32) simulácie, detailne analyzovať a zmapovať procesy konkrétnej reštaurácie a implementovať ich do simulačného softvéru. Táto práca si vyžiadala zhromaždenie reálnych dát priamo z prevádzky, konzultácie s jej vedením a validáciu (IMS, 37) modelu na základe reálnych prevádzkových podmienok. Výsledný model je unikátny svojím priamym uplatnením v konkrétnej reštaurácii.

1.3 Autori a zdroje informácií

Táto práca bola vypracovaná dvojčlenným tímom, ktorý tvorili: Jakub Hrdlička (xhrdli18) a Rebeka Tydorová (xtydor01). Informácie pre pochopenie základných princípov diskrétnej simulácie sme čerpali zo študijnej opory Modelování a simulace IMS (2022), ktorej autorom je Dr. Ing. Petr Peringer. Odborným konzultantom a dodávateľom faktov z oblasti gastronómie je prevádzkar podniku Iná haluška v Trnave – Rastislav Klačman. V jeho reštaurácii sme sa zastavili aj osobne a pozorovali chod prevádzky.

1.4 Prostredie overovania

Experimentálne overovanie modelu prebiehalo v simulačnom softvéri SIMLIB/C++ s reálnymi dátami o prevádzke reštaurácie (počet zákazníkov, časy obsluhy). Validita modelu bola kontrolovaná porovnaním výstupov so skutočnými štatistikami. Výsledky simulácie odpovedajú očakávanému chovaniu systému, čo potvrdzuje správnosť a použiteľnosť modelu.

2. Rozbor témy a použitých metód/technológií

Navrhovaný model simuluje prevádzku gastro zariadenia s otváracími hodinami od 10:00 do 21:30. Zameriavame sa na príchody zákazníkov, obsluhu, prípravu jedál a pohyb zákazníkov v reštaurácii. Údaje potrebné pre vytvorenie simulácie boli získané vlastným meraním v reštaurácii a zisťovaním u prevádzkara.

2.1 Popis použitých postupov

Pre generovanie časových intervalov medzi príchodmi zákazníkov sme použili rovnomerné rozdelenie (IMS, 89), čo umožňuje jednoducho nastaviť minimálny a maximálny interval pre každú časť dňa:

- Obedová fáza - 10:30 až 15:00, interval príchodov je 7 až 10 minút
- Poobedná fáza – 15:00 až 18:30, interval príchodov je 18 až 20 minút
- Večerná fáza – 18:30 až 21:30, interval príchodov je 10 až 13 minút

Použitie rovnomerného rozdelenia je jednoduché a vhodné pre potreby simulácie, pretože umožňuje flexibilné nastavenie intervalov bez nutnosti modelovania (IMS, 8) zložitejšej štatistiky.

Na pracovnom rajóne sa nachádzajú traja ľudia – kuchár, čašník a pomocník. Pomocník o 15:00 odchádza, do konca otváracej doby prevádzky ostávajú už len dvaja ľudia.

Príprava jedla trvá 6-7 minút a jeho konzumácia 15-20 minút. Obidve činnosti sú simulované pomocou rovnomerného rozdelenia. Čas potrebný na upratanie stola je 1 minúta.

63% zákazníkov konzumuje jedlo na mieste, 37% si jedlo berie so sebou. Toto rozhodnutie je simulované náhodným procesom s pevne danými pravdepodobnosťami.

2.2 Popis pôvodu použitých metód/technológií

Simulačný model je implementovaný v jazyku C++ pomocou knižnice Simlib, ktorá umožňuje jednoduché modelovanie procesov a správu front.

Rovnomerné rozdelenie bolo zvolené na generovanie časových intervalov a trvaní procesov z dôvodu jeho jednoduchosti a dostatočnej presnosti pre potreby simulácie. Umožňuje intuitívne nastavenie minimálnych a maximálnych hodnôt časov, čo je vhodné pre simulovanie procesov, kde je potrebné určenie rozsahu. Implementované bolo pomocou funkcie Uniform(a, b) zo Simlibu, kde ‘a‘ a ‘b‘ reprezentujú minimálny a maximálny čas.

Generovanie rozhodnutia zákazníka, či konzumuje jedlo na prevádzke, alebo si ho vezme so sebou, je realizované pomocou diskrétneho pravdepodobnostného rozdelenia, pretože ide o jednoduchý proces s fixnými pravdepodobnosťami. Táto metóda je efektívna a ľahko implementovateľná pomocou náhodnej funkcie na porovnanie s prednastavenými hodnotami.

Použité metódy nevyžadovali vytváranie nových algoritmov, pretože dostupné nástroje v rámci Simlib plne pokrývajú potreby simulácie.

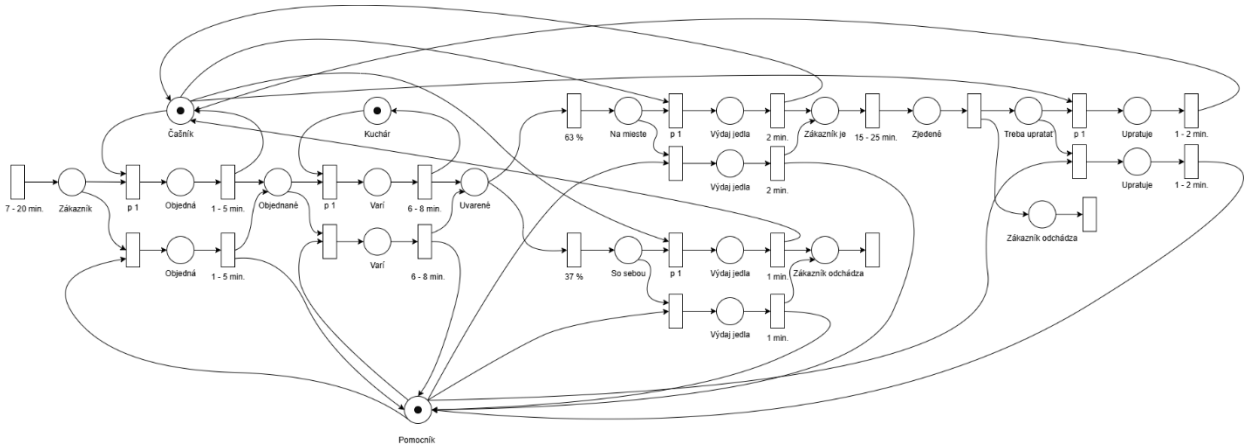
3. Koncepcie

3.1 Modelárske témy

Konceptuálny model predstavuje abstrakciu reštaurácie a redukuje jej fungovanie na základné procesy: príchod zákazníkov, objednávanie, príprava jedla, rozhodovanie o konzumácii, konzumácia a odchod. Zvoleným spôsobom znázornenia modelu je Petriho sieť (IMS, 126), ktorá:

- reprezentuje jednotlivé stavy (čakanie na čašníka, príprava jedla...) a prechody medzi nimi,
- synchronizuje procesy (dostupnosť čašníka či kuchára ovplyvňuje správanie zákazníkov).

Na obrázku je zobrazená abstraktná Petriho sieť reštaurácie:



Abstraktný model: Petriho sieť

3.2 Implementačné témy

Program je postavený na modelovaní prevádzky reštaurácie, kde sú jednotlivé procesy riadené prostredníctvom front a zariadení, ktoré reprezentujú dostupné zdroje. Hlavné časti programu sú:

- Zákazník - proces, ktorý reprezentuje jednotlivé kroky, ktorými zákazník prechádza počas svojho pobytu v reštaurácii
- Generátor zákazníkov – spúšťa procesy zákazníkov v pravidelných intervaloch
- Facility – predstavujú rôzne služby (čašník, pomocník, kuchár), ktoré spravujú interakcie so zákazníkmi
- Fronty – uchovávajú zákazníkov, ktorí čakajú na dostupnosť služieb

3.3 Princíp fungovania systému

Zákazníci prichádzajú do reštaurácie na základe časového intervalu. Na začiatku každého intervalu sa aktivuje nový proces zákazníka, ktorý vstupuje do fronty čašníka alebo pomocníka. Ak sú čašník a pomocník obsadení, zákazník čaká vo fronte. Ak sú voľní, zákazník je obslúžený a pokračuje v objednávaní jedla. Ak sú voľné aj zariadenia kuchár alebo pomocník, zákazník pokračuje do ďalšej fázy, kde čaká na prípravu jedla.

Po dokončení objednávky sa zákazník rozhodne, či bude jesť v reštaurácii, alebo si jedlo zoberie so sebou. Ak sa rozhodne pre konzumáciu na mieste, po uplynutí doby konzumácie zákazník odchádza a čašník alebo pomocník uprace stôl. V opačnom prípade zákazník odchádza po vydaní jedla.

4. Architektúra simulačného modelu/simulátoru

V simulačnom modeli sú zákazníci reprezentovaní triedou Customer. Každý zákazník je inštanciou tejto triedy, ktorá vykonáva rôzne činnosti v rámci simulácie. Hlavné procesy zákazníka sú nasledovné:

- order_food() - objednáva jedlo, ak je čašník alebo pomocník obsadený, čaká, a keď sa uvoľní, pokračuje ďalej,
- wait_for_food() - po objednaní čaká na prípravu jedla,
- decide_where_to_eat() - rozhodne sa, či bude jesť na mieste, alebo si jedlo vezme so sebou,

- take_order() - dostane jedlo, po podaní objednávky pokračuje v procese konzumácie alebo odchodu,
- eat() - ak bude jesť na mieste, konzumuje jedlo,
- clean() - po jedle zákazník odchádza, čašník alebo pomocník upratuje stôl, čo je modelované pomocou času cleanTimeMin až cleanTimeMax.

Generovanie zákazníkov je implementované pomocou triedy CustomerGenerator. Každý zákazník je vytvorený na základe náhodne generovaného intervalu medzi hodnotami arrivalMin a arrivalMax, ktoré definujú, ako často zákazníci prichádzajú do reštaurácie.

V reštaurácii sú tri hlavné zariadenia:

- čašník - obsluhuje zákazníkov pri objednávaní jedla a pri podávaní jedla,
- kuchár - pripravuje jedlo,
- pomocník - pomáha kuchárovi a čašníkovi.

Každé zariadenie má svoju vlastnú rolu a môže byť obsadené, čo znamená, že zákazník musí čakať, ak zariadenie nie je dostupné. V kóde sú zariadenia reprezentované objektmi typu Facility. Ak je zariadenie obsadené, zákazník čaká v príslušnej fronte na obsluhu.

Pre správu čakajúcich zákazníkov používame triedu Queue. Máme dve hlavné fronty – waiterQueue (zákazníci čakajúci na obsluhu) a cookQueue (zákazníci čakajúci na prípravu jedla).

Pre generovanie náhodných časových intervalov používame funkcie Uniform a Random, ktoré generujú náhodné hodnoty pre rôzne činnosti napr. pre čas objednania alebo konzumácie jedla.

5. Podstata simulačných experimentov a ich priebeh

Simulačné experimenty majú za cieľ analyzovať vplyv rôznych intervalov príchodov zákazníkov na efektivitu obsluhy a vyťaženosť personálu v reštaurácii. Prostredníctvom simulácie chceme zistiť, ako zmeny vo frekvencii príchodov ovplyvňujú plynulosť obsluhy, čakacie doby zákazníkov, vyťaženosť personálu a celkovú efektivitu prevádzky. Zameriavame sa na optimalizáciu počtu personálu, aby sa minimalizovali riziká zdĺhavých čakacích dôb a preťaženia zamestnancov, čo by mohlo viesť k zníženiu spokojnosti zákazníkov a zamestnancov.

Model je nevyhnutný na simulovanie rôznych scenárov v kontrolovaných podmienkach, kde môžeme flexibilne meniť parametre a sledovať ich dopad na systém. Takéto testovanie v reálnej prevádzke by bolo časovo, finančne náročné a neefektívne. Simulácia teda umožňuje efektívne a rýchle experimentovanie s rôznymi nastaveniami a predikciou správania systému.

5.1 Postup experimentovania

Pri experimentovaní sme nastavovali rôzne hodnoty intervalu príchodu zákazníkov, kde sme simulovali nízku, strednú a vysokú frekvenciu príchodov podľa nameraných časov v prevádzke. Tiež sme sledovali ako by sa situácia vyvíjala v prípade extrémnych frekvencií (veľmi nízkych/vysokých). Následne sme sledovali metriky ako sú vyťaženosť zamestnancov, maximálna čakacia doba zákazníkov vo fronte a počet neobslúžených zákazníkov (ak fronty prekročia kapacitu). Nakoniec sme vyhodnotili výsledky – snažili sme sa pomocou získaných dát identifikovať, či systém zvláda prichádzajúci objem zákazníkov bez výrazného predlžovania čakacích dôb.

5.2 Dokumentácia jednotlivých experimentov

1. Nízka frekvencia príchodov - zákazníci prichádzali každých 18 až 20 minút. Výsledky:
    - vyťaženosť personálu: čašník 27%, kuchár 17%, pomocník 1%.
    - fronty neboli využité

    Systém je schopný obslúžiť všetkých zákazníkov s nízkou záťažou, pomocník tu nie je potrebný.
2. Stredná frekvencia príchodov - zákazníci prichádzali každých 10 až 13 minút. Výsledky:
    - vyťaženosť personálu: čašník 43%, kuchár 25%, pomocník 5%.
    - fronty neboli využité

    Systém je schopný obslúžiť všetkých zákazníkov so strednou záťažou, pomocník stále nie je dostatočne využívaný.
3. Vysoká frekvencia príchodov - zákazníci prichádzali každých 7 až 10 minút. Výsledky:
    - vyťaženosť personálu: čašník 49%, kuchár 36%, pomocník 18%.
    - fronta na objednávky bola využitá s maximálnou dĺžkou 1 a maximálnym časom stráveným vo fronte 1,39 minúty

    Systém je schopný obslúžiť všetkých zákazníkov s vysokou záťažou, pomocník začína byť viac využívaný.

Extrémy – v reštaurácii Iná haluška sa momentálne tieto hodnoty nevyskytujú no dovolili sme si ich zahrnúť do experimentov.

4. Zákazníci prichádzali každých 0 až 7 minút. Výsledky:
    - vyťaženosť personálu: čašník 88%, kuchár 63%, pomocník 88 %.
    - fronta na objednávky bola využitá s maximálnou dĺžkou 9 (priemer 1,5) a maximálnym časom stráveným vo fronte 24,31 minúty (priemer 2,82)

- fronta na výdaj jedla bola využitá s maximálnou dĺžkou 2 (priemer 0,13) a maximálnym časom stráveným vo fronte 4,38 minúty (priemer 1,65)
- jeden zákazník nebol obslúžený

Systém nie je schopný obslúžiť všetkých zákazníkov, zamestnanci sa blížia k hranici svojej kapacity.

5. Zákazníci prichádzali každých 20 až 30 minút. Výsledky:
    - vyťaženosť personálu: čašník 19%, kuchár 12%, pomocník 1 %.
    - fronty neboli využité

    Systém je schopný obslúžiť všetkých zákazníkov s nízkou záťažou, pomocník tu nie je potrebný.

5.3 Závery experimentov

Simulačné experimenty preukázali, že systém reštaurácie je schopný efektívne obslúžiť zákazníkov pri nízkych až stredných frekvenciách príchodov, pričom pomocník je pri týchto scenároch minimálne vyťažený. Pri vysokej frekvencii príchodov sa vyťaženosť personálu zvyšuje a pomocník začína byť potrebný. Extrémne vysoká frekvencia príchodov odhalila, že systém nie je schopný obslúžiť všetkých zákazníkov, pričom personál dosahuje vysoké hodnoty vyťaženosti a vznikajú výrazné fronty.

Na základe experimentov môžeme potvrdiť, že aktuálny počet zamestnancov je adekvátny pre typické denné frekvencie príchodov. V prípade očakávania extrémnej záťaže je však potrebné uvažovať o zvýšení kapacity alebo zlepšení procesov, aby sa predišlo nadmernej vyťaženosti personálu a strate zákazníkov.

Validita výsledkov bola potvrdená prostredníctvom porovnania vstupných údajov so skutočnými podmienkami reštaurácie, pričom model odráža reálne procesy a pracovné časy zamestnancov. Simulačné scenáre boli navrhnuté tak, aby pokryli široké spektrum prevádzkových situácií a výsledky sú v súlade s očakávanými.

Simulačný model poskytol hodnotné poznatky, ktoré môžu byť využité na optimalizáciu prevádzky reštaurácie a lepšie plánovanie personálnych kapacít. Model je dostatočne flexibilný na ďalšie úpravy a rozšírenia, čo umožňuje jeho využitie aj v podobných prevádzkach.

6. Záver

Simulačné experimenty preukázali schopnosť modelu efektívne analyzovať prevádzku reštaurácie pri rôznych frekvenciách príchodov zákazníkov. Výsledky ukázali, že systém zvláda

nízke a stredné zaťaženie bez významného využitia pomocníka, zatiaľ čo pri vysokom a extrémnom zaťažení dochádza k nárastu vyťaženosti personálu, tvorbe front a predĺženiu čakacích dôb.

Validita modelu bola overená porovnaním so skutočnými podmienkami reštaurácie a výsledky zodpovedajú očakávaným prevádzkovým trendom. Model implementovaný v jazyku C++ pomocou knižnice Simlib umožňuje jednoduchú modifikáciu a rozšírenie na ďalšie scenáre.

V rámci projektu vznikol nástroj na analýzu prevádzky reštaurácie, ktorý môže byť využitý na plánovanie kapacít a optimalizáciu procesov. Tento nástroj poskytuje užitočné poznatky pre manažment prevádzky a ponúka základ pre ďalšie štúdie v oblasti simulácie gastro prevádzok.