Západočeská univerzita v Plzni

Fakulta aplikovaných věd

**Programovací techniky**

**semestrální práce 2015/16**

Petr Štechmüller, Antonín Vrba

**Standardní zadání semestrální práce pro PT 2015/2016**

Zadání je určeno pro **dva** studenty. Práce zahrnuje dvě dílčí části - vytvoření funkčního programu diskrétní simulace a napsání strukturované dokumentace.

**Zadání:**

Meziplanetární společnost "UFO-medical" zásobuje naši galaxii léky na vesmírný mor. Všech 5 zdrojových centrál, dokáže vyprodukovat dostatečné množství léků pro všechny planety, které zásobují. Zásoby jsou distribuovány jednotlivým planetám v závislosti na přijaté objednávce. Zásoby léků jsou pak rozváženy na měsíční bázi. Samozřejmě čas počítáme ve „hvězdných“ měsících, kdy má každý měsíc 30 dní (24 pozemských hodin), protože na každé planetě trvá lunární měsíc jinak dlouho.

Naše společnost zásobuje dohromady 5000 planet. Každá planeta má 100 tisíc až 10 milionů obyvatel. Jejich počet na jednotlivých planetách je určen normálním rozdělením pravděpodobnosti. Dá se ale říci, že nejvíce planet má kolem 3 milionů obyvatel. Každý obyvatel potřebuje k přežití 1 balení léku na měsíc. Každá planeta je schopna pokrýt měsíčně 20-80% zásob ze svých vlastních zdrojů. Toto číslo se ale každý měsíc mění, protože výroba léku není jednoduchá. Pokud tedy společnost nedokáže daný měsíc na planetu dovézt potřebné léky, tak zemře část obyvatel, pro které domácí produkce nestačí. Pokud počet obyvatel planety klesne pod 40 000, tak se stane pro společnost nezajímavou a přestane sem léky dovážet.

Galaxie má přibližnou rozlohu 800x800 LY (světelných let) a pro zjednodušení se dá zobrazit ve dvou rozměrech. Vzdálenosti mezi jednotlivými planetami jsou minimálně 2 LY. Mezi planetami vedou obchodní cesty. Každá planeta má obchodní cestu s pěti nejbližšími planetami, bohužel ne všechny cesty lze zabezpečit, a tak 20% cest hrozí útok vesmírných pirátů, kteří ukradnou veškerý náklad. Z dlouhodobých statistik společnost zjistila, že šance na přepadení lodi na nebezpečné cestě je 10%.

Společnost disponuje transportními loděmi, které jsou schopny převézt 5 milionů balení léku. Jejich rychlost je 25 LY/den a nakládka i vykládka nákladu trvá 1 den. Loď se po vyložení nákladu vrací do své mateřské základny.

Cílem je simulovat a řídit provoz zásobování planet tak, aby minimalizovali ztráty na životech (respektive maximalizovali zisky z prodeje léků).

**Vytvoření funkčního programu:**

* + Připravte rozumná vstupní data (centrály, planety, obchodní trasy, …) a uložte je ve vhodném formátu **(10b.)**,
  + zvolte a implementujte vhodné datové struktury pro reprezentaci vstupních dat, důsledně zvažujte paměťovou náročnost zvolených struktur a časovou náročnost algoritmů pro následovné výpočty **(10b.)**,
  + proveďte základní simulaci pro první den, vypište, která planeta je zásobována z které centrály, v jakém čase a množství, průběh simulace (všechny důležité hodnoty) zapisujte na obrazovku a do souboru **(10 b.)**.

**Výše popsaná část bude váš minimální výstup při kontrolním cvičení cca v polovině semestru.**

* + Vytvořte prostředí pro snadnou obsluhu programu (menu, ošetření vstupů) - nemusí být grafické, umožněte manuální zadání objednávky **(5b.)**
  + umožněte sledování (za běhu simulace) aktuálního stavu zásobování planety a sledování aktuálního stavu libovolné transportní lodě **(5b.)**,
  + proveďte simulaci pro 12 měsíců a vygenerujte do souborů následující statistiky (uložte je do vhodných souborů – **10b.**):
    - přehled jednotlivých planet s uvedením zavezeného množství léků rozepsané do jednotlivých měsíců, v rámci měsíců pak rozepsané dle jednotlivých transportních lodí. Dále vývoj počtu lidí na jednotlivých planetách v jednotlivých měsících. Na konci uveďte souhrnná čísla za celou dobu simulace a za každé čtyři měsíce simulace
    - přehled jednotlivých transportních lodí s uvedením jejich jednotlivých cest, převezených léků a počtu přepadení, na konci uveďte souhrnná čísla za celou simulaci
  + vytvořte dokumentační komentáře ve zdrojovém textu programu a vygenerujte programovou dokumentaci (Javadoc) **(10b.)**,
  + vytvořte kvalitní dále rozšiřitelný kód – pro kontrolu použijte softwarový nástroj PMD (více na http://www.kiv.zcu.cz/~herout/pruzkumy/pmd/pmd.html), soubor s pravidly pdmrules.xml najdete na portálu v pod menu *Samostatná práce* **(10b.)**
    - mínus 1 bod za vážnější chybu, při 6 a více chybách nutno opravit
    - mínus 2 body za 10 a více drobných chyb

**V rámci dokumentace:**

* + připojte zadání **(1b.),**
  + popište analýzu problému **(6b.),**
  + popište návrh programu (např. UML diagram) **(6b.),**
  + vytvořte uživatelskou dokumentaci **(5b.),**
  + zhodnoťte celou práci, vytvořte závěr **(2b.).**

**Poznámky:**

* + Příslušná data si vygenerujte.
  + Simulaci distribuce léků simulujte po dobu jednoho roku (12\*30 dní) po dnech.
  + Počet transportních lodí sice není omezen, lodě ale samovolně nezanikají, po vyložení celého nákladu se vrací nejkratší cestou zpět do mateřské centrály a jsou připraveny na další cestu.
  + Začínáte ve dni 1 prvního měsíce, předpokládejte, že k tomto dni mají planety dostatek léků na aktuální měsíc.
  + Planety odesílají objednávky na následující měsíc vždy začátku aktuálního měsíce (tedy objednávku na říjen odesílají 1. září).
  + Pokud společnost nestíhá doručit objednané léky do konce předcházejícího měsíce, tak je vůbec neodesílá (tj. objednávka na planetu nedorazí v půlce měsíce, během kterého byla potřeba).

# Analýza problému

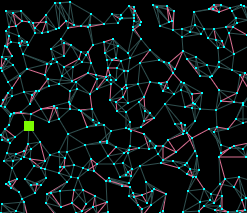
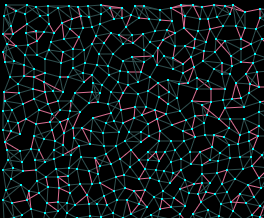
Naším úkolem bylo vytvořit simulaci dodávek léků po vygenerované galaxii za jeden rok. Problém po přečtení zadání se zdá být velice komplexní a tak, jsme se rozhodli programovat nejdříve jednotlivé moduly programu zvlášť a poté je složit do většího celku.

Výčet modulů je následující :

* Vygenerování galaxie (planety, stanice, cesty, konfigurační soubory)
* Algoritmus pro vyhledání nejkratší cesty
* Směrovací systém pro lodě (typy lodí, předávání lodí, logika dodávek)
* Podpůrné funkce (vizualizace simulace, logování, grafické rozhraní)

## Vygenerování galaxie

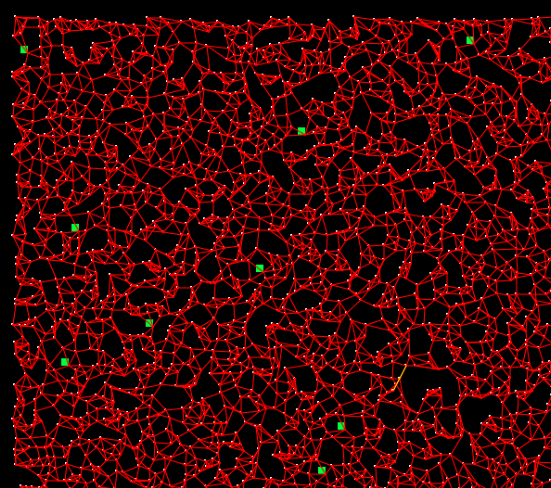
Představme si prostor 800 x 800 bodů, naše galaxie nabízí 640 tisíc potenciálních pozic pro planety a stanice, chtěli jsme systém udělat co nejvíce univerzální a tak nabízíme uživatelům vygenerovat galaxii různých parametrů a místo 5000 planet a 5 stanic si můžeme nechat vygenerovat galaxii o padesáti planetách a jedné stanici, tato konfigurace pak razantně snižuje obtížnost jakéhokoli následného debugování, které bylo během vývoje důležité. Díky skutečnosti, že planet je o mnoho méně než volných pozic, byl zvolen jednoduchý algoritmus, který postupně generuje souřadnice x, y a kontroluje zda na dané pozici není již obsazeno, pokud obsazeno je, tak se souřadnice jednoduše vygenerují znovu, tato metoda je při počtu planet do několika tisíc o mnoho rychlejší než kdybychom při každém vygenerování souřadnic kontrolovali celý list zatím vygenerovaných planet. Implementace je realizována přes „okupační“ matici typu boolean, kterou následně využíváme pro hledání sousedů a také pro kontrolu, že každá planeta má určitou vzdálenost od dalších planet. Při umístění hodnoty do okupační matice se nejen nastaví hodnota pro jednu souřadnici, ale také pro zadaný rádius kolem planety, který má možnost uživatel zadat. Vliv parametru „planet spacing“ viz Obr.1 (spacing = 2) a Obr.2 (spacing = 8). Při rozměru galaxie 800 x 800 a počtu planet 5000 generujeme přibližně 6 tisíckrát, což není vůbec špatné.

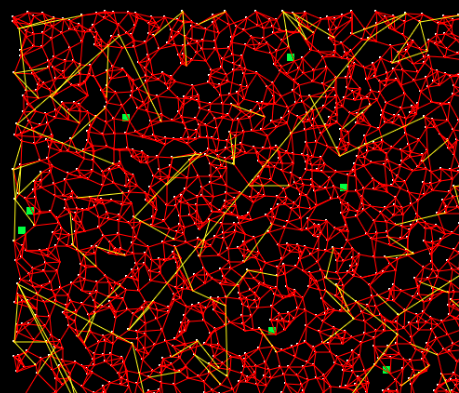
 

Obr.1

Obr.2

Vrcholy máme vygenerované, nyní je čas přiřadit každé planetě 5 cest k pěti nejbližším planetám. Nejdříve jsme se rozhodli pro primitivní neefektivní řešení, ke každému vrcholu vypočítáme cestu ke všem ostatním, kteří ještě nemají 5 sousedů, tuto množinu poté seřadíme a z vrcholu si vezmeme sousedů kolik potřebujeme. Tato metoda funguje, nicméně je neuvěřitelně pomalá a vyskytuje se zde zásadní problém, totiž poslední planety, si mohou nejbližšího souseda najít na druhé straně galaxie protože vše ostatní je již obsazené, vznikají ošklivé čáry přes celou galaxii, které by porušovaly konzistenci dat a zároveň zadání. Vymysleli jsme tedy nový algoritmus vyhledávání sousedů.

Osvětlíme si postup v několika krocích, každá planeta i stanice má společného předka *BaseObject*, ten má metodu „najdi si sousedy“. Při zavolání metody se kolem planety začne rozvíjet oblast, která se podle počtu potřebných sousedů dynamicky zvětšuje a přidává okolní planety do listu, pod podmínkou že mají volné místo pro souseda, když se jich najde dostatečný počet, seřadí se a nechají se vytvořit vazby sousedství. Tato samotná metoda neskutečně urychlí hledání sousedů, ale neřeší problém čar přes celou galaxii. Ten řeší až obslužná metoda hledání cest. Prakticky rozděluje celou galaxii na libovolné zóny, my jsme zvolili vodorovné pruhy určité výšky. Postupně zvětšujeme oblast směrem dolů a voláme metody „najdi si sousedy“, problém cest přes celou galaxii byl vyřešen. Výsledek je na Obr. 3 a 4 , žlutá cesta je delší než 30 jednotek.



A

Obr.4

Obr.3

## Algoritmus pro vyhledání nejkratší cesty

V tomto okamžiku jsme měli konzistentní neorientovaný graf připravený na další akce. Implementovali jsme Dijkstrův algoritmus pro hledání nejkratší cesty, použili jsme prioritní frontu. Třída *DijkstraAlgorithm* dostane v konstruktoru vrchol, ze kterého potom počítá nejkratší cesty do všech ostatních. Naše extrémně jednoduchá implementace algoritmu nám dovoluje přepočítat vzdálenosti z pěti centrál do 5000 planet za 30ms, přičemž do cílové planety vybereme jen nejkratší cestu z nejvhodnější stanice. Tím si vytváříme vlastní „směrovací tabulku“, kterou v průběhu simulace pro lepší efektivitu přepočítáváme v reálném čase podle potřeby. Shodli jsme se, že velitelství přepočítá cesty ke všem planetám, pokud počet přepadení piráty bude větší než 100. Procedura se poté provádí několikrát do měsíce, bohužel nemá na efektivitu dodávek nikterak velký vliv, protože každý měsíc se piráti přemisťují, na druhou stranu, pokud by se piráti přemisťovali jednou za rok, tak už po pěti měsících bude naše velitelství dodávat léky planetám s téměř 100% spolehlivostí.

## Směrovací systém lodí

Systém jak lodě cestují je asi nejsilnější stránkou naší práce, simulujeme totiž **reálný provoz** ze kterého můžeme maximalizovat výstupní informace. V jakýkoli okamžik známe přesnou polohu všech lodí a provoz v jakémkoli uzlu nebo cestě. Rozdělili jsme si den na 25 dílů a každý tento díl je jeden „tik / update“ pro všechny lodě, planety, stanice, cesty a velitelství. Po stisknutí tlačítka simulace začne tikat timer umístěný v hlavním konroléru a celá simulace se probudí k životu. Jak se simulace provádí si ukážeme v několika postupných krocích.

Na začátku každého měsíce posílají planety objednávky specifikující množství, každou objednávku zpracovává velitelství a její objednávkový „firewall“ který kontroluje zda máme ve směrovací tabulce záznam o dané planetě, dále zkontroluje zda jsme schopni ve zbývajícím čase měsíce se dostat na danou planetu a také prověří planetu jestli nemá méně jak 40 tisíc obyvatel. Při splnění všech podmínek je objednávka předána k následnému zpracování. Základna zpracovává 200 validních objednávek za jeden update, ale může jich zpracovávat i tisíce.

Zpracovaná objednávka se přiřadí základně ve které začíná cesta k cílové planetě. Stanice loď naloží množstvím požadovaného nákladu, toto nakládání trvá jeden den – 25 tiků. Výhodou našeho systému je, že můžeme mít více tipů lodí např. nákladní lodě s vysokou kapacitou nebo rychlé průzkumné lodě hledající piráty, díky bohaté interface struktuře je není problém doimplementovat. V současném stavu program pracuje se základní *SmallTransportShip* její kapacita je 5 milionů balení léků a tak, pokud přijde objednávka nad její kapacity musí vyrazit dvě lodě. Stanice disponuje několika „doky“ na lodě: zaparkované lodě, nakládající se lodě, lodě připravené k odletu a jiné pomocné listy. I zde má uživatel sílu měnit simulační podmínky parametrem „maximální počet lodí na jednu stanici“, lodě se vytvářejí až do určitého množství a poté se používají už jen vytvořené lodě, které se po návratu do stanice hlásí znovu do služby.

Když loď opustí stanici okamžitě se dostává do konfrontace s potencionální hrozbou pirátů na každé cestě, pokud je chycena je nucena se vrátit a pokud by šlo zásilku ještě stihnout, přidá se objednávka do fronty. Směrování se provádí mezi cestami a planetami, každá cesta si drží své lodě po určitý čas, který je ekvivalentní délce cesty, každá planeta drží lodě pouze jeden update, pokud je však planeta cílová, loď vykládá náklad celý den a poté se vrací. Všechna funkcionalita systému je vizualizována a dobře pozorovatelná.

## Podpůrné funkce

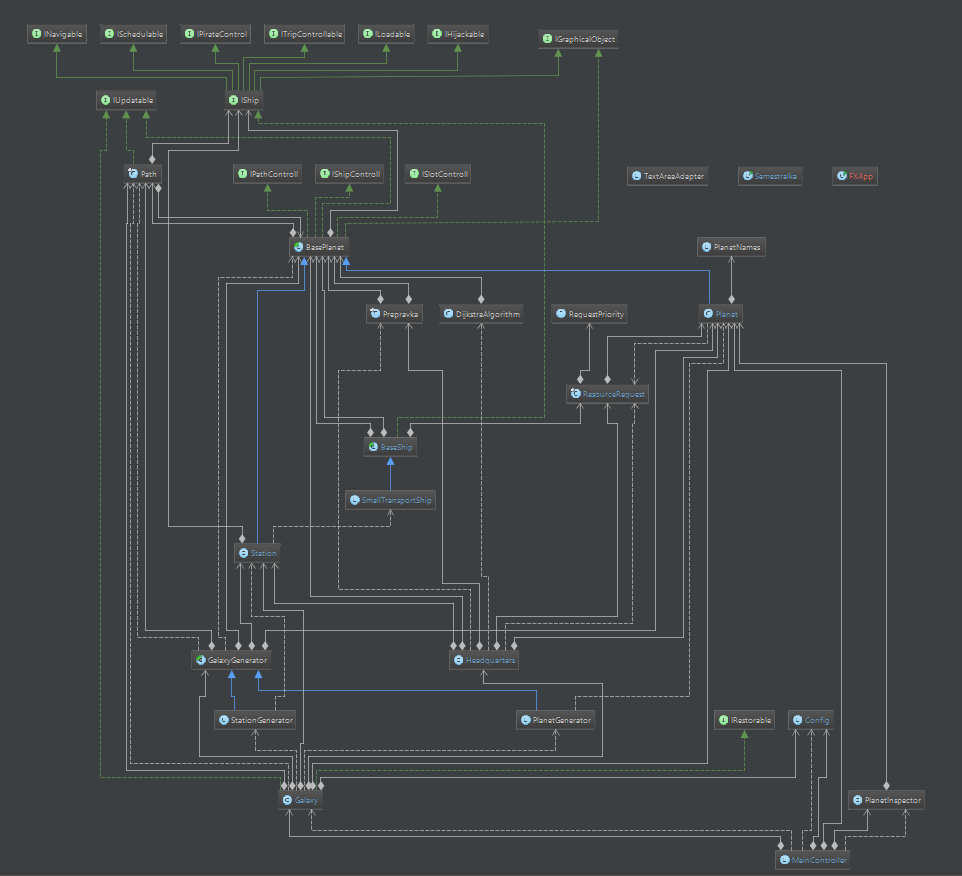
Pro logování různých výpisů používáme knihovnu log4j, která nabízí hned několik úrovní logování: error, info, debug. Grafické rozhraní disponuje textovou oblastí do které se vkládá právě úroveň info. Všechny úrovně jsou zároveň zaznamenávány do logovacího souboru. Aplikace každý měsíc zároveň generuje stav simulace a důležité hodnoty jako je např. počet mrtvých lidí nebo počet přepadení piráty za měsíc. Za celý rok se vygeneruje statistika stavu populace a dodávek na každé planetě. Pro grafické rozhraní jsme zvolili JavaFX, protože s ním máme dobré zkušenosti z předmětu UUR.

# Návrh programu

Program je realizovaný do několika packagů, kde v hlavním package je třída *Semestralka* s metodou *Main()*, která inicializuje FX aplikaci. Implementace generování galaxie můžeme naleznout v package s názvem *galaxy*, celá implementace Dijkstrova algoritmu je v package *dijkstra*. Takovou hlavní třídou je pomyslné „velitelství“ *Headquarters*  které řídí celou simulační proceduru a přijímá objednávky z planet.

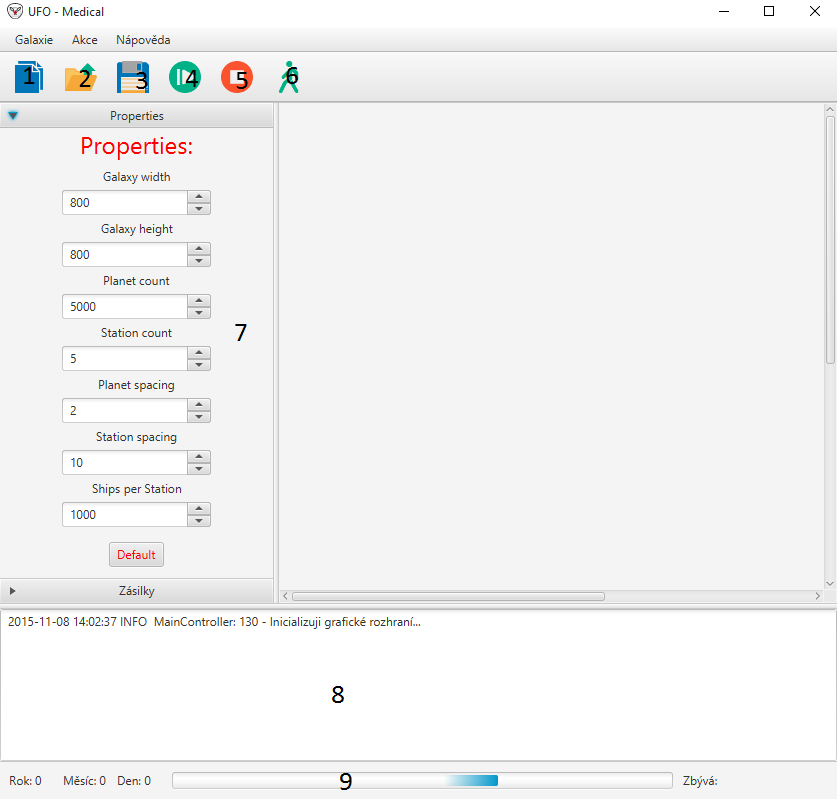
# UML diagram

Z důvodu Velkého UML diagramu přikládáme k dokumentaci UML soubor viz příloha, který lze otevřít např. v IntelliJ IDEA 14.



# Uživatelská příručka

Program se spouští přiloženým .bat souborem a je vyžadována instalace nejnovější verze Java. Po spuštění programu se zobrazí grafické rozhraní viz Obr.5. Uživatel může generovat (1) galaxii podle zadaných parametrů (7), pokud mu galaxie vyhovuje může si ji uložit (3) a při následném zapnutí programu přeskočit část generování a pomocí (2) importovat dříve uloženou galaxii. Stisknutí tlačítka generování není omezené a uživatel může v klidu zkoušet měnit parametry galaxie a znovu přepsat aktuální galaxii. Simulace se spouští pomocí tlačítka (4). Pro pozastavení simulace slouží tlačítko (5) a poté lze simulaci krokovat (6), pokračování simulace inicializuje opět tl. (4). Výstup z logování úrovně „info“ je zobrazeno v oblasti (8) a průběh simulace včetně znázornění jednoho roku je v sekci (9), simulace může běžět libovolně dlouho pro desítky let.



Obr.5

### Barevná vizualizace

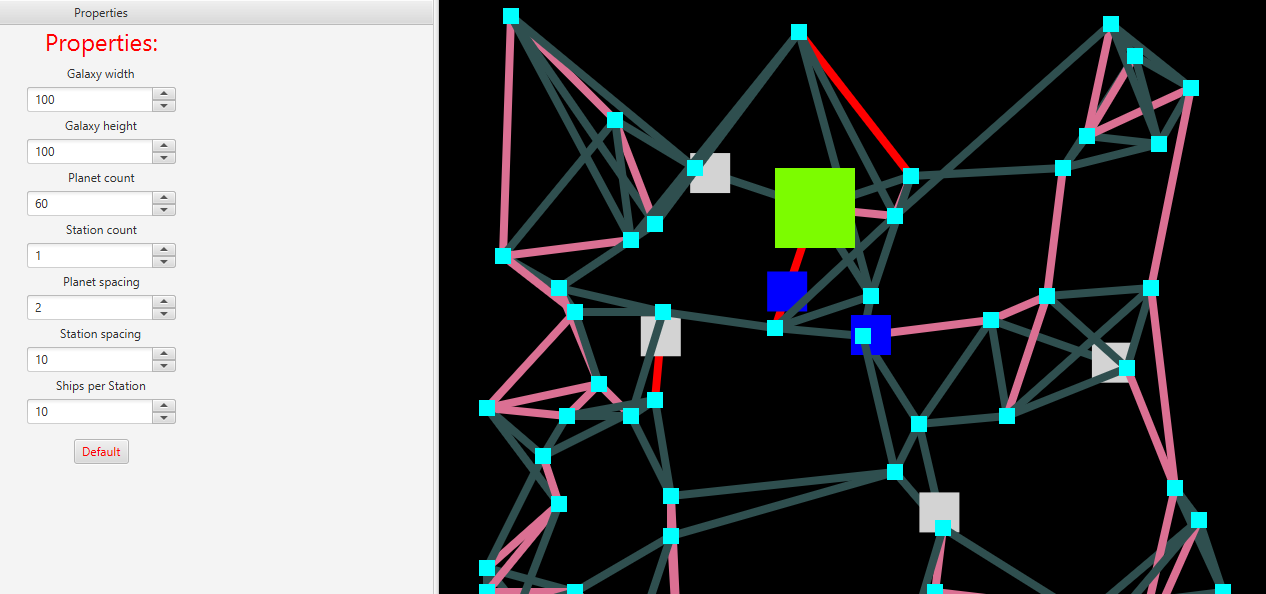
|  |  |  |
| --- | --- | --- |
| **Objekt** | **Barva** | **Popis** |
| Stanice |  | normální stav stanice |
| Cesta |  | cesta bez pirátů |
|  | potenciální hrozba pirátů |
|  | detekování pirátů |
| Planeta |  | planeta s více jak 40k ob. |
|  | nezajímavá planeta, méně jak 40k ob. |
| Loď |  | loď s nákladem vyslaná na cestu |
|  | loď se vrací prázdná (úspěch) |
|  | loď přepadli piráti a vrací se (neúspěch) |
|  | manuální objednávka (piráti jí ignorují) |

### Parametry simulace

Pokud chce uživatel konkrétní hodnoty parametrů tak po jejich změně je nutné v daném poli stisknout enter, aby se hodnoty propagovaly do konfigurace simulace. Nastavením parametrů lze změnit počet planet, stanic a prostor mezi nimi. Zajímavým nastavením je potom počet lodí na jednu stanici, které výrazně ovlivňuje stav populace v průběhu simulace. Pro ukázku zde uvedeme několik případů konfigurace galaxie, které povedou k naprosto odlišným výsledkům.

### Malá galaxie

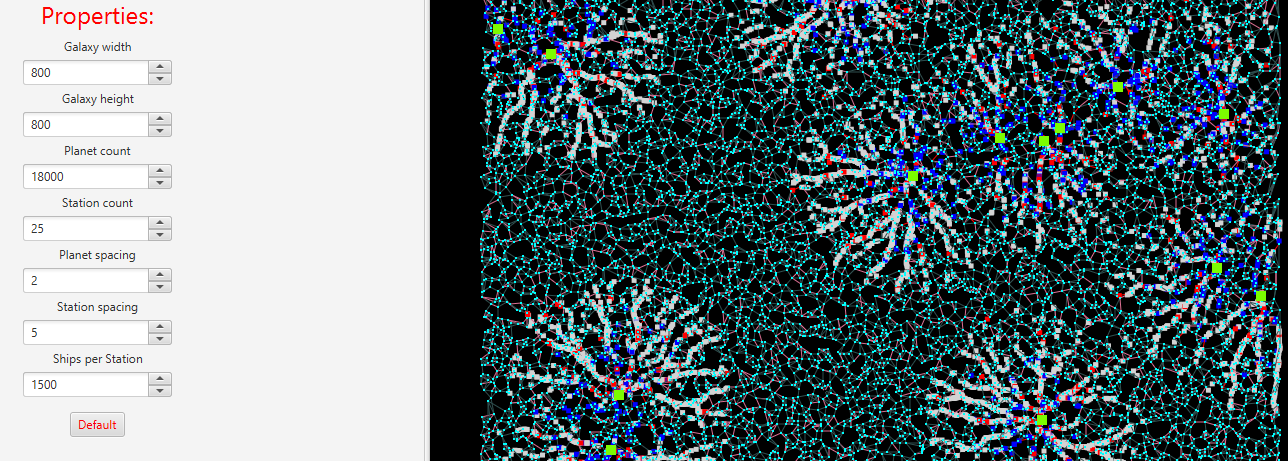
* + - Je vhodná pro testování logistiky a ladění programu.



Obr.6

### Velká galaxie

* + - Vhodná pro testování výkonnosti systému, na mapu 800 x 800 můžeme dostat 18 000 planet a celý systém se poté stará o cca 84 miliard lidí, i toto náš systém dokáže v reálném čase simulace vizualizovat.



Obr.7

### Manuální objednávky

Součástí je i modul obstarávající manuální dodávky k planetám, který lze otevřít kliknutím na roletu „zásilky“. Z aktualizovaného listu planet si můžeme libovolnou vybrat a poslat žádost o speciální druh objednávky kategorie „důležité“, tento typ objednávky je ve velitelském systému zpracování objednávek upřednostněn a loď která tuto objednávku veze je označena žlutou barvou. Doporučujeme tuto funkcionalitu testovat při malé mapě viz Obr. 6.

### Výstupní soubory

Program provádí logování do souboru „logging.log“ a na konci roku provede výpis ročních statistik planet (počet obyvatel, dodávky léků) do souboru „yearLog.log“. Oba tyto soubory lze nalézt v hlavní složce programu.

# Hodnocení práce

Zadání semestrální práce nebylo jednoduché a vyžadovalo hodně času na vypracování. Postupování po modulech se nám vyplatilo a když jsme programovali novou funkčnost věděli jsme, že stavíme na otestovaných a stabilních základech. Nejtěžší část nastala v okamžiku napojení vizualizace na směrovací systém a také směrování samotné nebylo vůbec jednoduché. Celé programování a ladění nám zabralo více než měsíc, kdy jsme se hodně naučili spolupracovat přes různé synchronizační programy, jedním z nich je platforma Floobits, která však s narůstajícím projektem začala dělat problémy a tak jsme se rozhodli pro jistotu ještě pravidelně verzovat na gitHub. Velkou zkušeností bylo i časté hledání problémů v rozsáhlém projektu zapříčiněné nejčastěji naprostou banalitou. Práce by šla ještě hodně vylepšit, ale na takové věci bohužel chybí motivace. S výslednou aplikací jsme spokojeni a většina věcí, které jsme si stanovili se nám podařilo implementovat.