Пловдивски Университет

Факултет "Математика и информатика"

Дипломна работа

Оптимизиране на транспортната мрежа в град Пловдив

Дипломант:
Венелин Вълков
Фак. № 0901261093

Научен ръководител:
гл.ас. Ангел Голев

03. 07. 2012 г.

Съдържание

Глава 1

Увод

Искали ли сте да прекарате повече време с детето си на закуска, да се приберете по-рано от работа и да прекарате време с любимият човек или да излезнете с приятели? Може и да поспите повече, разбира се. Колко хубаво би било това да е правило, вместо изключение. Всеки пловдивчанин, използващ пътната мрежа на града, губи средно около 60 мин. на ден в трафика.

1.1 Цел

Целта на дипломната работа е да предложи начин за намаляне на времето прекарано в пътната мрежа на град Пловдив с 10%.

1.2 Изисквания

- цената за използването на градската мрежа трябва да остане същата или да е по-ниска
- повишаване на комфорта и спокойствието при използването на градската мрежа
- увеличаване печалбите на превозващите компании
- намаляне на вредните емисии във въздуха
- намаляне на броя катастрофи

1.3 Постигане на целта

За постигане на целите на дипломната работа, спрямо поставените изисквания, се разглеждат -

- модерни технологични решения в подобни ситуации
- създаване на високо паралелизирана симулация за намиране на критичните точки от транспортната мрежа и тяхното оптимизиране
- за получените резултати се анализират за да се получи частично или пълно решение на проблема

1.4 Целеви групи

Хора, които използват транспортната мрежа, през най-натоварените часове на денонощието. Важно за всеки от участниците е бързото достигане на съответна точка от града, без това да пречи на личното им здраве и комфорт.

1.5 Желан резултат

Работата може да се сметне за успешна, ако се постигне намаляне на прекараното време в транспортната мрежа с 10%.

1.6 Задачи

- Проучване върху методите за изграждане на симулации
- Избор между съществуваща и специализирана за целта система
- Избор на програмен език
- Програмиране на самата система
- Провеждане на симулации

1.7 Структура

Настоящата дипломна работа се състои от:

Увод обосновка на проблема, поставяне на конкретна цел, целеви групи и желан резултат

Изисквания

Изложение разделено на три основни части:

Проучване основни концепции при създаване на компютърна визуална симулация

Създаване на симулация система, специфично създадена за нуждите на градската мрежа в град Пловдив. Избор на технологии. Програмиране.

Провеждане на симулации използване на реални данни като вход за създадената система

Резултати обявяване на получените резултати

Заключение Наблюдения върху получените резултати и дискусия. Постигнати ли са поставените цели и къде е имало проблеми. Какво може да бъде развито в бъдеще.

Глава 2

Изложение

2.1 Проучване

2.1.1 Какво е моделиране?

Моделирането е процесът по създаване на модел. Моделът е представяне на съществуващ обект от анализираната система. Той е близък, но по-прост от обекта в реалната система. Когато създава модел, анализаторът използва модела за да предскаже промените в системата. От една страна, моделът трябва да е близък до реалния обект и притежава неговите свойства. От друга страна, трябва да е лесен за разбиране и променяне. Добрият модел е добър компромис между леснота и реализъм [Anu]

Препоръчва се, усложняването на модела да става итеративно (iteratively). Важно е, моделът да продължава да бъде верен през този процес. Някои техники за това включват:

- Симулиране, включваики модела, с познати входни и изходни данни
- Пресечено валидиране (cross-validation) [Mahoney]

В зависимост от средствата, използвани за построяването им, моделите биват: физически (обекти, процеси и явления, евентуално различни по физическата си природа от оригинала, но с аналогични свойства), математически (теории, методи и обекти – функции, уравнения, редове и други), информационни (информационни методи, обекти и процеси), компютърни (програми, данни и други) и така нататък [Totkov]

Компютърното моделиране на информацията и автоматизирането на информационните дейности предполагат въвеждането, изучаването и използването на различни модели на информацията за обектите и за дейностите, в които те участват. Информационните обекти и информационните процеси са абстрактни

модели на информацията и информационните дейности, наречени абстрактни (концептуални) информационни модели. В компютърната информатика абстрактните информационни модели се проектират и създават под формата на алгоритми и структури от данни. [Totkov]

2.1.2 Какво е симулация?

Симулация е процес при който се извършват различни действия/експерименти върху построения модел, вместо върху реална система. В най-общото си значение, симулация е инструмент за наблюдаване на това как една система работи. Симулацията може да се тества под различни условия и за различни времеви диапазони.

Симулация може да се използва когато искаме да:

- Намаляне на риска за недостигане на поставен срок
- Намиране на непредвидени пречки и проблеми
- Оптимизация на поведението на системата

Видове:

Непрекъсната симулация(Continuous simulation) променя състоянието си в неопределени моменти във времето, използват диференциални уравнения

Дискретно-събитийна симулация(Discrete-event simulation) променя състоянието си в определени моменти във времето, използват се събития

2.1.3 Визуална симулация

Интересно за поставената цел е, че е необходима направата на визуален компонент за системата. По този начин, лесно ще може да бъдат наблюдавани различни движения на обектите. Това може да доведе до по-успешна и лесна оптимизация на цялостната пътна мрежа в града.

Такъв тип симулация налага използването на знания от компютърната графика. Още повече, доближава ни (и дори довежда) до използването на системи за съставяне на компютърни игри. Изненадващо, игрите не са разглеждани задълбочено, или поне тяхната структура и начин на работа, в академичните среди [Holzkorn].

Визуалната част на системата е критична за това дали системата ще бъде използвана. Лесното и удобно използване са важни за всеки софтуерен продукт. Потребителят е този, за който системата ни трябва да се грижи добре. [Microsoft]

Следва разглеждане на някои от основните и най-добри симулационни системи[1]. Това се прави с цел, правене на възможно най-добър избор на такава, която е в съответствие с поставените ни задачи и изисквания.

2.1.4 Преглед на системи за създаване на симулация

Simplex3

Simplex3 е система за симулации, която може да работи под Windows и Unix операционни системи. Тя е универсално приложима когато имаме дискретни модели, процеси лесно моделируеми с помощта на опашки или транспортни модели.

Как се създава симулация

Позволява лесно и бързо научаване на системата, като това не изисква научаването на нов програмен език от високо ниво.

Технологии

Притежава собствен език за създаване на модел, наречен Simplex-MDL. Той позволява описанието на почти всякакъв вид модели. Въпреки това, модели използващи частни диференциални уравнения не могат да се представят лесно.

Приложение

Благодарение на универсалността си, Simplex3 може да се използва за академични цели. Лесно може да се приспособи за случаи, в които няма разработен специализиран симулационен софтуер.

Оценка

От сайта на системата, не става ясно до колко тя е поддържана. Липсва добра документация.[Simplex3]

AnyLogic

AnyLogic е платена система за общ вид моделиране и симулационен инструмент за дискретни, непрекъснати и хибридни системи.

[1] `http://en.wikipedia.org/wiki/List_of_computer_simulation_software`

Как се създава симулация

Системата предоставя GUI(Graphical User Interface) за създаване на моделите, които допълнително могат да бъдат разширявани с помощта на програмният език Java. AnyLogic предоставя моделиране чрез UML-базирано обектно-ориентирано моделиране, блок-схеми, диаграми на автомати, диференциални и алгебрични уравнения и други.

Технологии

Предоставя възможност за създаване на Java аплети, които позволяват лесно споделяне на симулациите, както и поставянето им в интернет.

Програмни парадигми

Моделиращият език на системата е разширение на UML-RT - голяма колекция от най-добри практики доказали се при моделирането на големи и сложни симулации.

Приложение

Тя е полезна когато искаме да разглеждаме симулации относно:

- Контролни системи
- Производство
- Телекомуникации
- Обучение
- Логистика(Logistics)
- Компютърни системи

Оценка

AnyLogic е доста обширна и понякога тежка система. Предоставя голям набор от полезни инструменти. Тя е *много скъпа*[2], което е в разрез с изискванията ни. [AnyLogic]

[2]Цената на най-евтиното издание на продукта е 4800 €

Tortuga

Tortuga е симулационна система с отворен код, разработена от MITRE Corporation [3] в периода от 2004 до 2006 година. Тя е ориентирана изцяло към дискретно-събитийни симулации.

Как се създава симулация

Симулация в системата може да се представи като взаимодействия между процеси или планирани (scheduled) събития. Системата изисква множество от написани (от потребителя) класове на Java, инсталиран JDK (Java Development Kit), Ant - инструмент за автоматично билдване (build) и самата симулационна система. Предоставя се и GUI за управление и наблюдение на самата симулация.

Технологии

Tortuga е изцяло написана на Java. Това прави възможно използването ѝ под Windows и Unix базирани системи. Интересното тук е, че за постигане на своите цели Tortuga използва AOP (Aspect Oriented Programming) [AOP], като не се изисква от потребителя да има познания в тази насока, стига да компилира програмата си по специфичен начин.

Програмни парадигми

Системата използва програмна парадигма която многократно намаля сложността при създаване на симулация. Тя третира всяка единица от симулацията като отделна нишка. Java виртуалната машина ограничава броя на нишките [4], което води до ограничение броя на симулационните единици.

Приложение

Tortuga може да бъде използвана самостоятелно или като част от по-голям проект. Тя предоставя споделяне на симулации в Java аплети, също както и AnyLogic.

Оценка

[3] `http://www.mitre.org/`

[4] Броят варира спрямо имплементацията, настройки при пускане и хардуера

Лицензът който използва е LGPL[5], което прави системата добър кандидат за по-нататъшно разглеждане. Възможността й за използване под много платформи (благодарение на Java виртуалната машина) е още един голям плюс. Не трябва да забравяме използването и на *AOP*, което я изкарва от "стандартна"Java програма. Според сайта на системата, от 2008 година, поддръжката за нея е спряна. [Tortuga]

SimPy

SimPy е система с отворен код, създадена през 2002 г. от Klaus G. Muller и Tony Vignaux. След това към тях се присъединяват много други разработчици. Тя предоставя възможност за създаване на дискретно-събитийни симулации.

Как се създава симулация

Симулациите в системата могат да се представят като взаимодействия между процеси. Те се изграждат чрез програмен код. За създаването на проста симулация се изискват около 10 реда код, които включват познания по основни класове и функции от системата. Има GUI(използващо библиотеката Tk[6]) за предоставяне на входни данни и наблюдаване на симулациите.

Технологии

SimPy е написана на Python[7] и поддържа версии от 2.3 до 3.2 включително. Благодарение на Python виртуалната машина (при използване на стандартната CPython имплементация[8], SimPy може да се изпълнява под Windows и Unix операционни системи.

Програмни парадигми

Системата използва обектно-ориентиран подход към създаване на симулации. Благодарение на езика на който е написана, с нея бързо може да се направи дори и по-сложен модел.

Приложение

SimPy може да бъде използвана самостоятелно, както и навякъде където

[5] `http://www.gnu.org/licenses/lgpl.html`
[6] `http://www.tcl.tk/`
[7] `http://www.python.org/`
[8] `http://wiki.python.org/moin/CPython`

има Python интерпретатор. Предоставя се набор от пакети за изобразяване на графики и управление на структури от данни, които могат да бъдат използвани извън рамките на библиотеката. Готови симулации лесно могат да бъдат пакетирани като изпълними файлове и предоставени на други потребители.

Оценка

Лицензирана е под LGPL лиценз. Системата изглежда добре поддържана. Последната версия е 2.3, излезнала през Декември 2011 г. Използва лесен и достъпен език, но без много допълнителни инструменти за по-бързо и лесно създаване на симулация. [SimPy]

GarlicSim

GarlicSim е безплатен продукт с отворен код[9], който се опитва да предостави нова концепция, за това как всъщност може да се използват симулациите. Автор на системата е Ram Rachum[10].

Как се създава симулация

Проектът обещава да напише повтарящата се част от кода, необходим за една симулация, вместо нас и ни остави да работим върху по-важната част от проекта - самата симулация. Самата тя, се създава чрез писане на програмен код. Може да постигнем направата на проста симулация с 5 реда код!

Технологии

GarlicSim е изцяло написана на Python и официално все още не поддържа Python 3.x сериите. Системата предлага изчистен и лесен GUI за управление и наблюдение на симулациите.

Програмни парадигми

За създаване на системата е използван изцяло обектно-ориентиран подход, който допринася за лесното научаване на основните стъпки при правене на симулация. Основна дейност при извършване на симулация е т.нар. симулационен пакет (simulation package), който съдържа функция която определя стъпката за дадената симулация.

[9] `https://github.com/cool-RR/GarlicSim`
[10] `http://ram.rachum.com/`

Приложение

Според автора, системата е достатъчно обща за да позволи симуларинето на каквато и да е симулация. Дадени възможности са:

- физични
- теория на игрите
- разпространение на епидемии
- квантова механика
- електрически

Оценка

Лицензирана е под LGPL лиценз. GarlicSim е един чудесен продукт за начални опити за създаване на симулации. Основен проблем е, че е в алфа версия. Няма сведения за успешни употреби в някаква насока. Има добра, но не напълно достатъчна документация. За създаване на собствена симулация е необходимо написването на симулационен пакет, което до някъде обезсмисля обещанието за това да пишем по-малко код. [GarlicSim]

Глава 3

Заключение

Библиография

[AOP] Yuliyan Kiryakov and John Galletly. Aspect-oriented programming – case study experiences. 2003.

[Anu] Maria Anu. Introduction to modeling and simulation.

[AnyLogic] `http://www.xjtek.com/anylogic/why_anylogic/`.

[GarlicSim] `http://garlicsim.org/`.

[Holzkorn] Peter Holzkorn. Physics simulation in games. 2008.

[Mahoney] Kevin Mahoney. Model validation techniques. 2010.

[Microsoft] Microsoft Corporation. Usability in software design. 2000.

[SimPy] `http://simpy.sourceforge.net/`.

[Simplex3] `http://www.simplex3.net/Body/Introduction/English/indexAbstract.html`.

[Tortuga] `http://code.google.com/p/tortugades/`, 2004.

[Totkov] Георги Тотков. Концептуално и компютърно моделиране на езикови структури и процеси (с приложения за българския език). 2004.