1. Въведение

Този проект се фокусира върху проблема за създаване на N-Body симулация. Дава се описание на задачата, както и математическа формулировка. Извършва се анализ на системата, като се описва желаната функционалност, анализират се съществуващи алгоритми както и екземпляри които използват тези алгоритми. На база на този анализ се избират подходящи алгоритми, които да се използват в системата. След фазата анализ се описва разработката на проекта, анализ на получените резултати, описва се структурата на проекта, дават се описания за употреба на системата и насоки за бъдещо развитие.

* 1. Описание на задачата

Същността на N-Body симулацията е да се проследят взаимодействията между n на брой тела. Тези взаимодействия могат да се подчиняват на закони като гравитацията или други физични явления. Този проект се фокусира върху взаимодействията които се ръководят от гравитацията. Този случай е полезен за астрономията, защото позволява да се симулира начина, по който тела във вселената се движат.

Крайната цел на проекта е да се създаде симулация на n на брой тела, с еднаква маса като се сравни последователната и паралелната обработка.

* 1. Математическа Формулировка

Тази секция ще разгледа N-Body проблема от математическа гледна точка. Основната цел е да се обяснят формулите, които се използват. Има две основни категории, изчисления за привличането и изчисления за движението на частиците.

* + - Изчисления за привличане
    - Изчисления за движение

1. Анализ на системата
   1. Желана Функционалност
   2. Анализ на алгоритми
      * Наивен (Pairwise interaction)

Този алгоритъм е най-простия за разбиране и имплементация. При него изчисленията са всяко със всяко. Поради тази причина има сложност O(N2) и при нарастване на броя частици квадратично нараства времето за изчисление на един кадър от симулацията. За научни симулации това е голям недостатък, но за този проект това е голямо предимство, защото може лесно да се види разликата между последователната и паралелната обработка.

* + - Barnes Hut

Този алгоритъм е по-сложен от предходния, но има няколко предимства.

* + - Други
    - Алгоритми за придвижване на частиците
  1. Анализ на екземпляри
  2. Избор на алгоритми

Въпреки, че всички екземпляри използват алгоритми различни от Pairwise interaction, този проект го имплементира заедно с BH алгоритъма.

Това решение беше взето, защото при наивния метод може лесно да се покаже какво влияние има паралелната обработка, докато BH алгоритъма може да покаже как последователни части в паралелна програма оказват влияние върху паралелизма на програмата.

1. Реализация
   1. Избор на език и среда за разработка

За език проекта използва C#, а средата на за разработка е Visual Studio 2019. Използвам C#, защото с него имам най-много опит и езика има добри възможности за паралелизъм.

Друга причина да използвам C# е възможностите за визуализация и взаимодействие с потребителите. За създаване на интерфейса използвам WPF, а за визуализацията използвам библиотеката SkiaSharp за генериране на отделните кадри на симулацията и библиотеката Xade.FFMPEG използвам за генериране на видео от отделните кадрни.

Интерфейса се реализира с помощта на WPF(Windows Presentation Foundation).

* 1. Описание на имплементираните алгоритми
     + Pairwise interaction (Наивен подход)
     + BH (Barnes-Hut)
  2. Описание на реализацията
     + Кратко описание

Реализацията на проекта се разделя на две основни части, интерфейса и библиотека (class library). Интерфейса съдържа единствено логиката за обработка на входните данни, създаване на инстанция на основния клас от библиотеката (ProgramManager), извикване на функции от този клас и визуализиране на информация отнасяща се до статуса на изпълняващата се операция. Комуникацията между библиотеката и интерфейса е чрез извикване на функции (от страна на интерфейса) и генериране на събития (events) от страна на библиотеката.

Благодарение на това разделение, библиотеката може да се използва в бъдеще отделно от сегашния интерфейс. Секцията с диаграмите ще покаже нагледно описаната структура с пример за споменатите събития.

Раздела „Упътване за употреба“ има подобно описание на интерфейса и всички функции.

* + - Генериране на частиците

Програмата разполага с пространство за симулация с размери по X: 737px, по Y:979px. При генериране на частиците, се създава матрица с размерите на симулационното пространство, където всяка „клетка“ представлява един пиксел.

Генерирането позициите на частиците е произволно и при създаване на всяка една частица се извършва проверка дали няма частица на тази позиция с помощта на матрицата, ако има се генерира нова позиция.

Това е необходимо, за да не се генерират две частици в една и съща точка, защото това би нарушило структурата на дървото, което се използва при BH алгоритъма.

* + - Симулация

В тази точка, ще опиша процеса на изчисляване и визуализиране на един кадър от симулацията.

* + - 1. Необходими изчисления

За да се създаде един кадър трябва да се направят следните изчисления:

* Изчисляване на силата на привличане между двата обекта
* Изчисляване на разстоянието между два обекта
* При BH алгоритъма се проверява дали разстоянието между частица и клетка е повече от два пъти дължината на страната на клетката. Като в случая, поради правоъгълната форма на клетките като „дължина на страна“ се взима по-дългата страна.
* Изчисляване движението на частицата. Това е изчислението на позицията, която частицата ще заеме в следващия момент от време.
  + - 1. Стъпки

Изчисленията описани в предходната точка трябва да се изпълнят в определена последователност. Диаграмата на ***<ADD FIGURE NUMBER>*** показва стъпките за създаване на симулация използвайки наивния алгоритъм а на ***<ADD FIGURE NUMBER>***се показват стъпките при създаване на симулация използвайки BH алгоритъма.

* + - 1. Генериране на резултати

Резултатите от симулацията са .png изображения с размера на симулационното пространство (737х979). Запазват се в директория като се именуват с поредни номера.

* + - Генериране на видео
      1. Обосновка

Реших да генерирам кадри и видео вместо да показвам симулацията в „реално време“ защото поради ограничения на технологията за интерфейса, е невъзможно да се направи гладка симулация и в същото време тази симулация да се запази във файл. Тъй като симулацията е единствения начин, по който човек може да се види резултата от кода за мен беше важно крайния резултат да е гладка анимация, която да е 60 кадъра в секунда.

* + - 1. Библиотеката FFMPEG

За генериране на видеото от поредица от изображения използвам външната библиотека Xade.FFMPEG, която е адаптация на библиотеката FFMPEG за C#. Реших да използвам външна библиотека, защото прецених, че задачата за генериране на видео е извън обхвата на този проект.

С помощта на тази библиотека с един ред код всичките генерирани изображения се преобразуват във .mp4 видео което може да се гледа в 60 кадъра в секунда.

* + - Автоматично тестване

За улеснение и точност тестването на програмата е автоматично.

Автоматичното тестване измерва за колко време се изчислява един кадър от симулацията при различен брой нишки.

Автоматичното тестване може да се настрои да тества един от двата алгоритъма, колко нишки да могат да се използват, вида на нишките (дали са ръчно направени или се използва TPL библиотеката на C#) и колко пъти се повтаря даден тест за да филтрира ефекта на фоновото натоварване.

* + - 1. Измерване на време

Времето за изпълнение се измерва с вградения клас на C# „Stopwatch”. Таймера се пуска точно преди да започнат изчисленията за един кадър и се спира веднага след края. Времето за рисуване не се отчита, защото при автоматично тестване не се рисуват кадрите.

* + - 1. Генериране на резултати

Резултатите от автоматичното тестване са време за изпълнение, ниво на паралелизъм и ефективност. Данните се визуализират на три отделни графики. По-подробна информация може да се види в раздел 4 „Резултати“

* 1. Структура на програмата
     + Обща диаграма
     + Диаграми за симулацията
     + Диаграми за автоматично тестване

1. Резултати
   1. Тестов план
   2. Получени резултати
   3. Анализ
2. Упътване за употреба
   1. Описание на интерфейса
   2. Видове настройки
   3. Стъпки за употреба
      * Стъпки за симулация
      * Стъпки за автоматично тестване
3. Насоки за бъдещо развитие

Поради ограниченото време и липсата на опит в разработването на подобни системи, има още какво може да се подобри в програмата.

Може да се направи по-добра паралелна имплементация на Barnes-Hut която да (Morton curves?)

Може да се увеличи мястото за симулация

1. Източници

Въведение/описание на проблема

Видове алгоритми за nbody

Анализ на източници (Morton Curves)

Избран алгоритъм + обоснование

Наивен (последователен/паралелен)

BH (последователен/паралелен)

Обосновка за избор на технологии

Начин на реализация

Детайли относно алгоритмите

Диаграма на програмата

Резултати

Какво измервам

Анализ на резултатите

Инструкции за употреба

Инсталация

Как да се използва (user guide)