



МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»
ФАКУЛЬТЕТ ІНФОРМАТИКИ ТА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ
КАФЕДРА ІНФОРМАТИКИ ТА ПРОГРАМНОЇ ІНЖЕНЕРІЇ

Курсова робота з дисципліни «Моделювання систем»

Тема: Імітаційна модель роботи регулювальної ділянки цеху
на основі формалізму мережа Петрі

Керівник:

асистент кафедри ІІІ
Дифучина Олександра Юріївна
«Допущено до захисту»

(особистий підпис керівника)

«__» _____ 2023 р.

Захищено з оцінкою

Члени комісії:

Виконавець:

Чапча Святослав
Олександрович

студент IV курсу

групи ІТ-04

залікова книжка № ІТ-0425

«25» грудня 2023 р.

Інна СТЕЦЕНКО

Олександра ДИФУЧИНА

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Кафедра інформатики та програмної інженерії
Дисципліна «Моделювання систем»
Спеціальність Інженерія програмного забезпечення
Курс 4 Група ІТ – 04 Семестр 7

ЗАВДАННЯ

на курсову роботу студента
Чапчи Святослава Олександровича
(прізвище, ім'я, по батькові)

1. Тема роботи: Імітаційна модель роботи регулювальної ділянки цеху на основі формалізму мережа Петрі

2. Термін здачі студентом закінченої роботи "25" грудня 2023 р.

3. Вихідні дані до проекту

Завдання № 10 з Навчального Посібника

4. Зміст розрахунково-пояснювальної записки (перелік питань, що розробляються)

Вступ. 1. Розробка концептуальної моделі 2. Розробка формалізованої моделі 3. Програмна реалізація моделі 4. Проведення експериментів 5. Інтерпретація результатів експериментів. Висновки. Список використаних джерел. Додатки.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
Графічного матеріалу не має.

6. Дата видачі завдання "12" вересня 2023 р.

РЕФЕРАТ

Курсова робота: 64с., 12 рис., 6 табл., 2 додатка, 7 джерело літератури.

Об'єкт дослідження – регулювальна ділянка цеху.

Мета роботи – визначення оптимального режиму роботи регулювальної ділянки цеху.

Метод дослідження – імітаційне моделювання роботи регулювальної ділянки. Проведено дослідження різних режимів роботи регулювальної ділянки цеху і розроблена програмна реалізація імітаційної моделі системи. Розроблено план і проведені експерименти з імітаційною моделлю. Результати моделювання використані для визначення оптимального режиму роботи регулювальної ділянки цеху.

Ключові слова: ІМІТАЦІЙНА МОДЕЛЬ, МОВА МОДЕЛЮВАННЯ, ПЛАНУВАННЯ ЕКСПЕРИМЕНТІВ, ДИСПЕРСІЙНИЙ АНАЛІЗ, РЕГУЛЮВАЛЬНА ДІЛЯНКА ЦЕХУ, МЕРЕЖА ПЕТРІ

ЗМІСТ

ВСТУП	6
ПОСТАНОВКА ЗАВДАННЯ.....	8
РОЗДІЛ 1. КОНЦЕПТУАЛЬНА МОДЕЛЬ	9
1.1 Дослідження можливих видів моделювання	9
1.2 Ціль моделювання.....	10
1.3 Концептуальна модель	11
1.4 Вхідні та вихідні дані, параметри моделі	11
1.5 Обмеження.....	12
1.6 Цільова функція	13
РОЗДІЛ 2. ФОРМАЛІЗОВАНА МОДЕЛЬ.....	14
2.1 Елементи мережі Петрі.....	14
2.2 Побудова формалізованої моделі	14
2.3 Обчислення вихідних характеристики	15
РОЗДІЛ 3. РЕАЛІЗАЦІЯ МОДЕЛІ.....	17
3.1 Опис програмної реалізації імітаційної моделі	18
3.2 Оцінка адекватності моделі	19
3.3 Верифікація моделі	21
РОЗДІЛ 4. ПРОВЕДЕННЯ ЕКСПЕРИМЕНТІВ НА МОДЕЛІ.....	23
4.1 Постановка задачі	23
4.2 Тактичне планування.....	23
4.3 Стратегічне планування	24
РОЗДІЛ 5. ІНТЕРПРЕТАЦІЯ РЕЗУЛЬТАТІВ МОДЕЛЮВАННЯ....	28
ВИСНОВОК	29

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	30
Додаток А. Лістинг коду	31
Додаток Б. Результати проведення експериментів.....	44

ВСТУП

В Україні цехи є важливим елементом економіки. Вони забезпечують робочими місцями тисячі людей та виробляють значну частину промислової продукції. У зв'язку з війною та іншими геополітичними факторами зростає потреба в локальному виробництві товарів та послуг. Цехи можуть допомогти забезпечити цю потребу.

Метою дослідження є визначення оптимального режиму роботи регулювальної ділянки цеху.

Для того, щоб вирішити поставлену задачу, потрібно створити модель роботи регулювальної ділянки. Для цього ми використаємо метод імітаційного моделювання.

Модель - це абстрактне відображення об'єкта, системи або концепції, призначене для наукового вивчення. У сучасному світі практично неможливо знайти галузь людського життя, де не використовуються різноманітні методи моделювання.

В загальному випадку модель складається з X - множини вхідних змінних системи, Y - множини вихідних змінних системи, P - множини параметрів та F - функції, функціоналу, алгоритму або формального представлення залежності змінних Y від змінних X .

Різні методи моделювання використовуються у всіх сферах людської діяльності і є невід'ємною частиною нашого життя. Основна мета моделювання полягає в тому, щоб знайти значення вихідних змінних Y при відомих значеннях вхідних змінних X , враховуючи відому модель F та визначені параметри P .

Імітаційне моделювання – це метод, який дозволяє досліднику отримати інформацію про властивості реальної системи, багаторазово запускаючи її модель. Імітаційне моделювання враховує зміну властивостей об'єктів у часі, тому їм можна вирішити будь-яке завдання.

Основна перевага імітаційного моделювання – це можливість розробки моделей з мінімальною витратою часу на програмування. Також імітаційне моделювання дозволяє вирішувати складніші завдання, ніж аналітичне дослідження, оскільки воно враховує випадкові дії та інші фактори. Алгоритми імітації мереж Петрі та мереж масового обслуговування, які побудовані на основі універсальних мов програмування, мають високу гнучкість.

Імітаційне моделювання складається з таких етапів:

1. Формулювання проблеми та змістовної постановки задачі;
2. Розробка концептуальної моделі;
3. Розробка імітаційної моделі;
4. Оцінка адекватності моделі;
5. Планування та проведення експерименту;
6. Оцінка точності результатів моделювання;
7. Інтерпретація результатів моделювання і прийняття рішення.

Для розв'язання поставленої задачі ми побудуємо концептуальну модель, формалізовану модель із формалізмом мережа Петрі, які відображатимуть роботу регулювальної ділянки. Також ми виконаємо програмну реалізацію моделі зі вказаним формалізмом та проведемо перевірку й експерименти.

ПОСТАНОВКА ЗАВДАННЯ

На регулювальну ділянку цеху через випадкові інтервали часу надходять по два агрегати в середньому через кожні 30 хвилин. Первинне регулювання здійснюється для двох агрегатів одночасно і займає біля 30 хвилин. Якщо в момент приходу агрегатів попередня партія не була оброблена, агрегати на регулювання не приймаються. Агрегати, які одержали відмову, після первинного регулювання надходять у проміжний накопичувач. З накопичувача агрегати, що пройшли первинне регулювання, надходять попарно на вторинне регулювання, яке виконується в середньому за 30 хвилин, а ті, що не пройшли первинне регулювання, надходять на повне регулювання, що займає 100 хвилин для одного агрегату. Всі величини задані середніми значеннями, розподілені за експоненціальним законом.

Визначити ймовірність відмови в первинному регулюванні і завантаження накопичувача агрегатами, що потребують повного регулювання. Визначити параметри і ввести в систему накопичувач, що забезпечує безвідмовне обслуговування агрегатів, що надходять.

РОЗДІЛ 1. КОНЦЕПТУАЛЬНА МОДЕЛЬ

1.1 Дослідження можливих видів моделювання

Щоб вирішити поставлене завдання, потрібно створити модель системи. Існує багато різних методів моделювання, але найпопулярнішими є аналітичне, імітаційне та статистичне моделювання [1].

Аналітичне моделювання є найстарішим і найпростішим методом моделювання. Воно використовується, коли залежність між вхідними та вихідними змінними системи можна описати аналітичними функціями. Це означає, що можна знайти формули або рівняння, які визначають, як зміняться вихідні змінні системи в залежності від змін вхідних змінних.

Унікальність аналітичного моделювання полягає в тому, що воно дозволяє отримати точніші результати, ніж інші методи моделювання. Це пов'язано з тим, що аналітичні функції зазвичай є точними апроксимаціями реальних систем. Однак аналітичне моделювання має і свої обмеження. Воно може бути застосоване лише до систем, для яких залежність між вхідними та вихідними змінними можна описати аналітичними функціями. Якщо така залежність не існує, то аналітичне моделювання неможливо використовувати.

Імітаційне моделювання є найскладнішим методом моделювання. Воно використовується, коли систему моделюють шляхом її імітації в часі. Цей метод дозволяє враховувати випадковість та інші фактори, що неможливо зробити за допомогою аналітичних або математичних методів [2].

Унікальність імітаційного моделювання полягає в тому, що воно дозволяє моделювати системи, які є занадто складними або непередбачуваними для інших методів моделювання. Це пов'язано з тим, що імітаційне моделювання дозволяє моделювати систему в її реальному середовищі, з урахуванням всіх випадкових факторів, але воно може бути застосоване лише до систем, для яких можна

розробити модель імітації. Якщо така модель не може бути розроблена, то імітаційне моделювання неможливо використовувати.

Статистичне моделювання - це метод, який використовується для створення математичної моделі ймовірнісного розподілу даних. Ця модель може використовуватися для прогнозування майбутніх даних, аналізу даних або для отримання інформації про дані. Статистичне моделювання є унікальним методом моделювання, оскільки воно дозволяє враховувати випадковість. Це означає, що статистичні моделі можуть бути використані для моделювання систем, які є занадто складними або непередбачуваними для інших методів моделювання [3].

З переваг статистичного моделювання варто зазначити, що воно враховує випадковість, має широкий спектр застосувань та статистичні моделі можна тестувати на основі даних, які не використовувались для навчання. Але статистичне моделювання також має і недоліки: воно залежить від даних, статистичні моделі не можуть гарантувати точність своїх прогнозів і вони можуть бути складними для інтерпретації.

1.2 Ціль моделювання

На регулювальну ділянку цеху через випадкові інтервали часу надходять по два агрегати в середньому через кожні 30 хвилин. Первинне регулювання здійснюється для двох агрегатів одночасно і займає біля 30 хвилин. Якщо в момент приходу агрегатів попередня партія не була оброблена, агрегати на регулювання не приймаються. Агрегати, які одержали відмову, після первинного регулювання надходять у проміжний накопичувач. З накопичувача агрегати, що пройшли первинне регулювання, надходять попарно на вторинне регулювання, яке виконується в середньому за 30 хвилин, а ті, що не пройшли первинне регулювання, надходять на повне регулювання, що займає 100 хвилин для одного агрегату.

Головною ціллю є визначити ймовірність відмови в первинному регулюванні і завантаження накопичувача агрегатами, що потребують повного регулювання.

Також потрібно визначити параметри і розмір накопичувача, щоб було забезпечено безвідмовне обслуговування агрегатів, що надходять.

1.3 Концептуальна модель

Концептуальна схема моделі, згідно з постановкою задачі, має вигляд, зображений на рисунку 1.1.

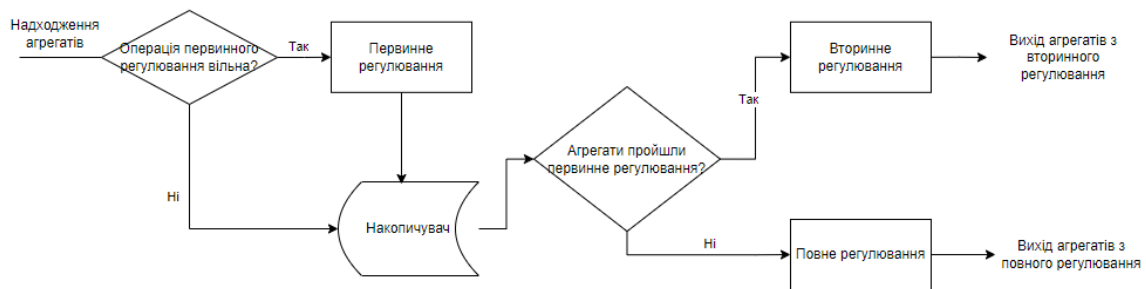


Рисунок 1.1 – Концептуальна модель

1.4 Вхідні та вихідні дані, параметри моделі

Вхідні та вихідні дані та параметри моделі, згідно з постановкою задачі, зображені у таблицях 1.1-1.2

Таблиця 1.1 – Вхідні параметри моделі

Параметри	Значення	Опис
T1	30(exp)	Періодичність поступання агрегатів
T2	30(exp)	Час здійснення первинного регулювання
T3	30(exp)	Час здійснення вторинного регулювання
T4	100(exp)	Час здійснення повного регулювання
N1	2	Кількість агрегатів, що поступають до регулювальної ділянки кожні T1 одиниць часу

N2	2	Кількість агрегатів, що поступають до первинного регулювання і обробляються T2 одиниць часу
N3	2	Кількість агрегатів, що поступають до вторинного регулювання і обробляються T3 одиниць часу
N4	1	Кількість агрегатів, що обробляються T4 одиниць часу на повному регулюванні.
Q	1440	Час моделювання

Таблиця 1.2 – Вихідні параметри моделі

Параметри	Опис
Q1 mean	Середня довжина черги оброблених первинним регулюванням у накопичувачі
Q1 max	Максимальна довжина черги оброблених первинним регулюванням у накопичувачі
Q2 mean	Середня довжина черги необроблених первинним регулюванням у накопичувачі
Q2 max	Максимальна довжина черги необроблених первинним регулюванням у накопичувачі
M1	Кількість агрегатів з вторинного регулювання
M2	Кількість агрегатів з повного регулювання
P	Ймовірність відмови в первинному регулюванні

1.5 Обмеження

Для системи існують наступні обмеження:

- $T1 > 0$
- $T2 > 0$
- $T3 > 0$

- $T_4 > 0$

1.6 Цільова функція

В нашому випадку ціллю моделювання є визначення ймовірності відмови в первинному регулюванні та завантаження накопичувача агрегатами. Таким чином цільова функція матиме вигляд:

$$P = M_{skipped} / (M_{skipped} + M_{processed}) \rightarrow \min$$

Інша ціль нашого моделювання – це визначення оптимального розміру накопичувача, при якому буде забезпечено безвідмовне обслуговування агрегатів. Для цього дізнаємось ймовірність того, що агрегат залишиться в накопичувачі та теоретично виміряємо якого розміру має бути накопичувач для безвідмовної роботи регулювальної ділянки.

РОЗДІЛ 2. ФОРМАЛІЗОВАНА МОДЕЛЬ

2.1 Елементи мережі Петрі

Після того, як ми створили концептуальну модель, нам потрібно визначити, як її формалізувати. Для цього існує два основних способи: мережі Петрі та мережі масового обслуговування.

Мережі Петрі - це потужний інструмент для моделювання дискретних процесів, які мають складні взаємозв'язки. Вони використовуються, коли потрібно моделювати систему з спільними ресурсами, які обслуговують багато процесів, або коли потрібно моделювати паралельні процеси. Важливою перевагою мереж Петрі є їхня універсальність, яка дозволяє моделювати як процеси управління, так і функціонування об'єктів управління [6].

На зображенні 2.1[4] показані компоненти формалізації мережі Петрі.

Е Л Е М Е Н Т И М Е Р Е Ж І П Е Т Р І




Перехід		позначає подію
Позиція		позначає умову
Дуга		позначає зв'язки між подіями та умовами
Маркер(один)		позначає виконання (або не виконання) умови
Багато фішок		позначає багатократне виконання умови
Багато дуг		позначає велику кількість зв'язків

Рисунок 2.1 – Елементи мережі Петрі

2.2 Побудова формалізованої моделі

Генератор приблизно кожні 30 хвилин створює по два агрегати і після цього, агрегати потрапляють у позицію Р2. Якщо первинне регулювання вільне, то агрегати попарно обробляються приблизно 30 хвилин і після цього потрапляють до накопичувача. Якщо первинне регулювання зайняте, агрегати також потрапляють в накопичувач. З накопичувача агрегати які обробились потрапляють у вторинне регулювання, якщо воно вільне вони обробляються приблизно 30 хвилин, а якщо ні, вони просто чекають обробки. З накопичувача агрегати, які не обробились потрапляють у повне регулювання, якщо воно вільне один агрегат обробляється 100 хвилин, а якщо зайнятий, агрегати просто чекають обробки. Одиницею модельного часу є 1 хвилина.

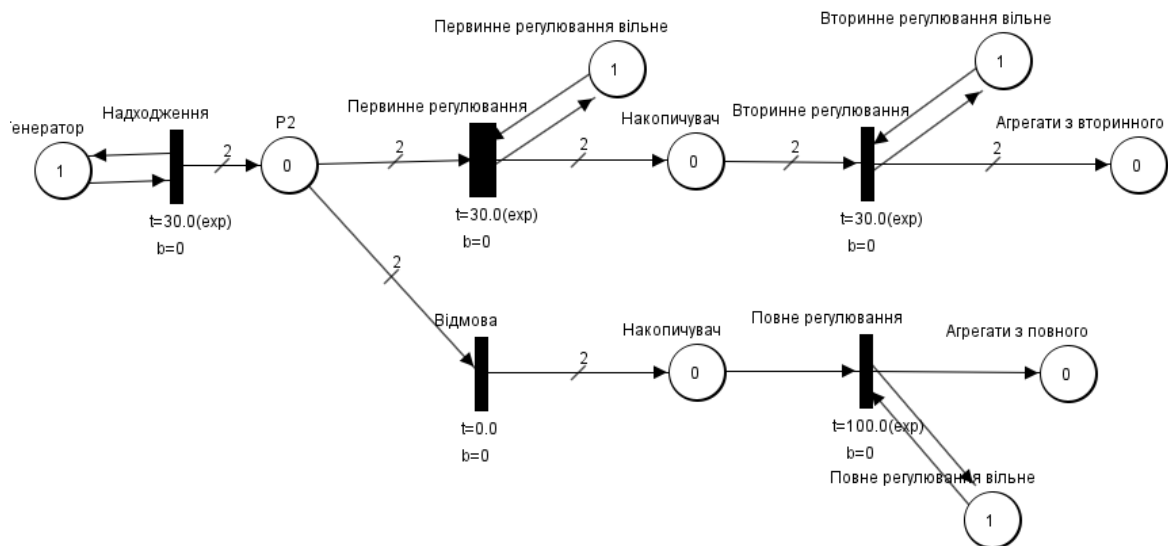


Рисунок 2.2 – Схема мережі Петрі, що відповідає моделі

2.3 Обчислення вихідних характеристики

Визначення середньої довжини черги агрегатів у накопичувачі:

$$Q_{mean} = \frac{\sum_{k=1}^n (Q1_k + Q2_k) \cdot \Delta t_k}{T_{mod}} \quad (2.1),$$

де $Q1$ – черга оброблених первинним регулюванням агрегатів у накопичувачі, $Q2$ – черга необроблених первинним регулюванням агрегатів у

накопичувачі T_{mod} – час моделювання, Δt_k – зміна часу під час k -того спостереження

Визначення ймовірності пропуску первинного регулювання:

$$P = \frac{M_{skipped}}{M_{skipped} + M_{processed}} \quad (2.2),$$

де $M_{skipped}$ – кількість пропусків первинного регулювання, $M_{processed}$ – кількість обробок первинним регулюванням.

РОЗДІЛ 3. РЕАЛІЗАЦІЯ МОДЕЛІ

Для програмної реалізації даного підходу було обрано програмне забезпечення PetriObjectModelPaint. Це програмне забезпечення має вбудовані засоби генерації випадкових величин за експоненційним, рівномірним і нормальним законами розподілу. Крім того, до коду програми можна додавати додаткові спеціальні класи та методи, що дозволяє підлаштувати програмну реалізацію конкретної задачі та розробити функціонал для підрахунку статистичних даних, виведення звітів у зручному форматі та автоматизованого багаторазового проведення експериментів з ітеративним змінням параметрів моделювання.

Програмне забезпечення має графічний інтерфейс, що дозволяє створити та візуалізувати модель. Можливість збереження візуального представлення як методу (у вигляді програмного коду) значно спрощує програмну реалізацію поставленої задачі. Крім того, програмне забезпечення дозволяє зберегти модель у вигляді файлу, що дозволяє проаналізувати модель багаторазово, не створюючи саму модель заново.

Програмне забезпечення для моделювання Петрі-об'єктів розроблено на мові Java[8]. Воно складається з пакету PetriObjLib, який реалізує алгоритм симуляції Петрі-об'єктів, і пакетів, що забезпечують графічне представлення мережі. Якщо мережа Петрі містить перехід без вхідних або вихідних місць, виникне помилка.

Після підготовки списку Петрі-об'єктів та визначення зв'язків між ними модель може бути створена за допомогою класу PetriObjModel. Метод go(double time) цього класу ініціює моделювання. Якщо генератор затримки поверне від'ємне значення, виникне помилка.

Основними завданнями програмного забезпечення є забезпечення правильного алгоритму моделювання та правильних результатів, включаючи

середні значення маркерів у місцях мережі Петрі, середнє значення буферів у переходах та стан мережі Петрі в кінці моделювання.

Для вирішення задачі були розроблені додаткові основні методи, які наведені у Додатку А.

Таблиця 3.1 – Основні методи

Назва функції	Опис
CreateAdjSectionNet	Метод, що створює об'єкту регульовальної ділянки цеху
getModel	Метод, що збирає Петрі-об'єкти в модель і виконує прив'язку
showStatistics	Метод, який проводить експерименти і виводить статистику з отриманих даних. Також метод включає у собі цільові функції та дисперсійний аналіз
validateModel	Метод, в якому ми порівнюємо теоретичний і практичний час надходження та обробок
chebishevExperiment	Метод, в якому ми знаходимо достатню кількість експериментів для подальшого аналізу
timeExperiment	Метод, в якому ми проводимо експеримент з часом для майбутньої побудови графіку залежності
firstExperiment	Метод, в якому модель спрацьовує один раз і виводить статистику з отриманих даних

3.1 Опис програмної реалізації імітаційної моделі

Проведемо пробний експеримент завдяки створеному раніше методу firstExperiment з наступними вхідними даними:

Таблиця 3.2 – Вхідні дані у пробному експерименті

Параметри	Значення
-----------	----------

T1	30(exp)
T2	30(exp)
T3	30(exp)
T4	100(exp)
N1	2
N2	2
N3	2
N4	1
Q	1440

Рисунок 3.1 – Результат пробного експерименту

Як можемо побачити, за час моделювання в один день (1440 хвилин) 48 агрегатів пройшли вторинне регулювання, 15 агрегатів пройшли повне регулювання, 40 агрегатів залишились у накопичувачі. Відсоток того, що первинне регулювання зайняте у час надходження становить 53%, а середнє завантаження проміжного накопичувача становить 38.8%.

3.2 Оцінка адекватності моделі

Використаємо створений раніше метод `validateModel`, який порівнює теоретичні та практичні значення часу надходження, часу обробки первинним регулюванням, вторинним регулюванням і повним регулюванням.

```

~Model Validation~
Theoretical T1 = 30(exp); Factual T1 = 34.285714285714285
Theoretical T2 = 30(exp); Factual T2 = 34.285714285714285
Theoretical T3 = 30(exp); Factual T3 = 34.285714285714285
Theoretical T4 = 100(exp); Factual T4 = 144.0

```

Рисунок 3.2 – Результат оцінки адекватності моделі

Виконаємо пробний експеримент з минулого пункту ще раз, щоб оцінити модель на адекватність.

```

Solo Experiment
MaxQ1: 10
MeanQ1: 1.219979713547636
LeftQ1: 0
MaxQ2: 32
MeanQ2: 14.059646787612325
LeftQ2: 30
NumOfFull: 17
NumOfSecond: 58
Failure Prob: 0.44761904761904764
Storage Load: 0.13390139797726022
BUILD SUCCESSFUL (total time: 0 seconds)

```

Рисунок 3.3 – Результат пробного експерименту

Як можемо побачити, за 1440 хвилин 58 агрегати пройшли вторинну обробку, 17 агрегатів пройшло повну обробку та 30 агрегатів залишились у накопучивачі, тобто усього 105 агрегатів надійшло на регулювальну ділянку цеху.

Перевіримо час надходження агрегатів з часом 30 хвилин:

$$\frac{1440}{30} * 2 = 96 \quad (3.1)$$

Як можемо побачити, практичне та теоретичні значення майже співпадають. Також обрахуємо приблизний час надходження агрегатів:

$$\frac{1440}{(17+58 + 30) \div 2} = 27.42 \quad (3.2)$$

Також зробимо це для обробки агрегатів у кожному регулюванні. Для вторинного регулювання:

$$\frac{1440}{58} = 24.83 \quad (3.3)$$

Для первинного регулювання:

$$\frac{1440}{30+0} = 48 \quad (3.4)$$

Для повного регулювання:

$$\frac{1440}{17} = 84.7 \quad (3.5)$$

Як можемо побачити, теоретичні значення майже співпадають з практичними, тому можемо зробити висновок, що програма функціонує правильно. Отже, модель є адекватною.

3.3 Верифікація моделі

Зробимо верифікацію моделі завдяки методу showStatistics. Для цього ми створимо кілька різних наборів вхідних значень і порівняємо результати.

Таблиця 3.3 – Набори вхідних значень для експериментів

Час надходження	Час обробки пер.рег.	Час обробки втор.рег.	Час обробки пов. рег.
10	60	20	20
100	30	30	100
40	40	40	140
30	30	30	100
20	20	20	80

Для генерації верифікаційної таблиці, проведемо експерименти поступово змінюючи параметри часу надходження, обробки та розмір вільного ресурсу у первинному, вторинному та повному регулюванні.

Create Delay	First Delay	Second Delay	Full Delay	First Queue	Second Queue	Full Queue	Max Q1	Mean Q1	Left Q1	Max Q2	Mean Q2	Left Q2	Num Of Second	Num Of Full	Failure Probability	Storage Load
10	60	20	20	1	1	2	2	0.0909	0	83	34.94	82	42	136	0.8384615	0.31923
100	30	30	100	2	2	1	2	0	0	2	0.02	0	26	2	0.0714286	0.07143
40	40	40	140	3	1	1	30	14.078	28	2	0.21	0	60	2	0.0222222	0.02222
30	30	30	100	1	1	1	4	0.292	0	40	17.7	40	44	15	0.5555556	0.40404
20	20	50	80	1	3	2	2	0	0	50	17.2	50	66	28	0.5416667	0.34722

Рисунок 3.5 – Верифікація моделі

Як можемо побачити, якщо час первинного регулювання більший за час надходження, то відсоток відмови буде більшим, як і завантаження накопичувача, а коли час надходження більший за час первинного регулювання, відсотки будуть майже нульовими. Також при збільшенні ресурсів, при однаковому часі, відсоток також буде нульовим, адже агрегати майже і не будуть потрапляти до повного регулювання.

Це відповідає очікуванням, тому модель є адекватною.

РОЗДІЛ 4. ПРОВЕДЕННЯ ЕКСПЕРИМЕНТІВ НА МОДЕЛІ

4.1 Постановка задачі

Основна ціль моделювання даної системи є визначення відсотку відмови первинного регулювання, якщо агрегати прийшли у момент, коли регулювання вже зайняте. Також необхідно визначити приблизний розмір накопичувача, щоб програма працювала безвідмовно.

Значення цільової функції – мінімізація ймовірності пропуску первинного регулювання, а також розрахування приблизного розміру накопичувача. Експерименти будемо проводити послідовно, задаючи різні вхідні дані.

4.2 Тактичне планування

Проведемо пробний прогін завдяки методу `timeExperiment` та побудуємо графік залежності ймовірності пропуску первинного регулювання від часу моделювання.

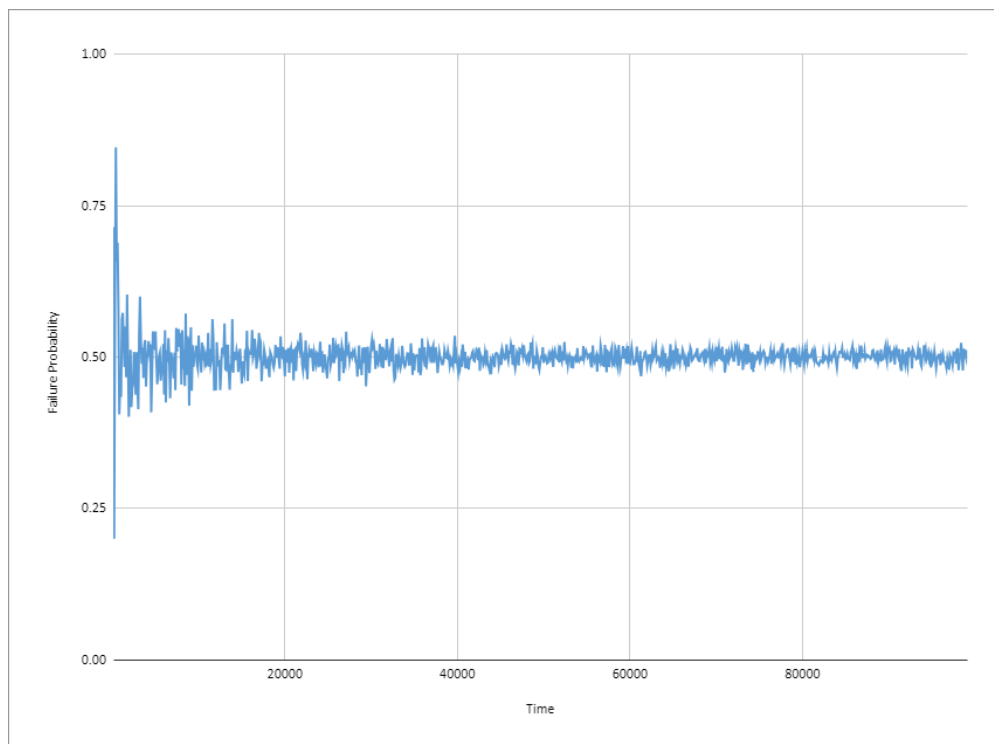


Рисунок 4.1 – Графік залежності ймовірності пропуску первинного регулювання від часу моделювання

На рисунку 4.1 можемо побачити, що в цілому час моделювання не сильно впливає на роботу моделі, але після приблизно 40000 од. часу графік більш стабільний та коливається у діапазоні від 40 до 60 відсотків. Тому візьмемо тривалість прогону в 40000 одиниць модельного часу для наступних експериментів, щоб перехідні процеси мало впливали на результат. Для визначення кількості необхідних експериментів за нерівністю Чебишева, скористаємося формулою:

$$N = \frac{\sigma^2}{\varepsilon^2(1-\beta)} \quad (4.1)$$

N – кількість прогонів, ε – точність оцінки (5% від середнього значення), β - довірна ймовірність (0.95), σ – дисперсія.

Скористаємось згаданим у третьому розділі методом `chebishevExperiment` для пошуку достатньої для заданої точності кількості експериментів відповідно до формули (4.1).

```

Chebyshev Test
Failure Probability = 0.4990702274168806
Standard Deviation = 0.01397925292419724
Enough number of experiments by Chebishev: 6.276746476926137

```

Рисунок 4.2 – Результат пошуку мінімальної кількості експериментів

4.3 Стратегічне планування

Для дослідження впливу факторів на значення цільової функції – ймовірність пропуску первинного регулювання, проведемо факторний експеримент.

Згідно з результатами експериментів Чебишева для кожної комбінації заданих параметрів потрібно провести 6 прогонів. Можливі комбінації зазначені у таблиці 4.1

Таблиця 4.1 – Набори вхідних значень для експериментів

Час надходження	Час обробки пер.рег.	Час обробки втор.рег.	Час обробки пов. рег.
20	20	20	60
30	30	30	100
40	40	40	140

Далі проведемо тестування для всіх комбінацій часу надходження та обробки. Для кожного з цих варіантів також проведемо по 50 запусків. Результати цих запусків наведені в додатку Б.

На підставі отриманих даних з додатку Б проведемо аналіз дисперсії. Суть дисперсійного аналізу полягає в тому, що ми порівнюємо дисперсію, яка виникає внаслідок впливу факторів, з дисперсією, яка пояснюється випадковими причинами. Якщо різниця між цими видами дисперсії велика, то фактори справді впливають на відгук моделі.

Спочатку, обрахуємо середні значення за формулами[4]:

$$\underline{y_j} = \frac{1}{p} \sum_{i=1}^p y_{ij} \quad \underline{y} = \frac{1}{q} \sum_{j=1}^q \underline{y_j} \quad (4.2),$$

Де p – кількість прогонів у кожному експерименті, q – рівень фактора, а y_{ij} – ймовірність пропуску первинного регулювання j -ого експерименту в i -тому спостереженні

Далі введемо величини[4]:

$$S_{\text{факт}} = p \cdot \sum_{j=1}^q (\underline{y_j} - \underline{y})^2 \quad S_{\text{залиш}} = \sum_{j=1}^q \sum_{i=1}^p (y_{ij} - \underline{y_j})^2 \quad (4.3),$$

Наступним кроком буде визначити загальну, факторну та залишкову дисперсії[4]:

$$d_{\text{факт}} = S_{\text{факт}} \quad d_{\text{залиш}} = \frac{S_{\text{залиш}}}{q \cdot (p-1)} \quad (4.4),$$

Далі знаходимо критерій Фішера[4]:

$$F = \frac{d_{\text{факт}}}{d_{\text{залиш}}} \quad (4.5)$$

```

Analysis Of Variance
Sum of Squares: 1682857.6685424028
Degrees of Freedom: 405.0
Mean Square: 693.7443903207102
F: 2425.760398241832
F-critical (alpha=0.05): 1.41805107
Influence is worthy of attention

```

Рисунок 4.3 – Отримані результати проведення дисперсійного аналізу

Як можемо побачити, критерій Фішера 2425.7604 при критичному значенні 1.418, а це означає, що вплив на режим роботи регулювальної ділянки є значним.

Тепер використаємо отримані дані для того щоб порахувати оптимальний розмір накопичувача, щоб регулювальна ділянка працювала безвідмовно. Для цього ми порахуємо кількість агрегатів які пройшли вторинне та повне регулювання та максимальну кількість агрегатів які були у накопичувачі. Завдяки цьому ми порахуємо відсоток на те, що агрегати залишаться у накопичувачі.

Для того, щоб віднайти оптимальний розмір накопичувача, я порахував теоретичний час надходження за формулою (3.1) і помножив його на знайдений раніше відсоток. Після цього, результат було округлено.

```

double total = probabilities.stream().mapToDouble(Double::doubleValue).sum();
double failureProbability = total / probabilities.size();
double storageLoad = (double) totalSumInStorageMean / (totalSumOfStucked + totalSumOfProcessed);
double storageAvg = (double) totalSumInStorage / (numOfSituations * numOfExp);
double stuckProbability = (double) totalSumOfStucked / (totalSumOfStucked + totalSumOfProcessed);
long adviceForSize = (long) ((Math.round(((totalSumOfStucked + totalSumOfProcessed) / (numOfSituations * numOfExp)) * stuckProbability) + 5) / 10) * 10;

```

Рисунок 4.4 – Пошук оптимального розміру накопичувача

Скористаємось згаданим у третьому розділі методом showStatistics для того, щоб порахувати оптимальний розмір накопичувача.

```

Main Goals
Failure Probability: 0.5002626756290645
Storage Load: 0.1832901276327242
Storage Avg Max: 1060.4115226337449
Stuck Probability : 0.36737123339593675
Advice For Size : 1060

```

Рисунок 4.5 – Результат пошуку оптимального розміру накопичувача при часу моделювання 40000хв

Як можемо побачити, якщо регулювальна ділянка буде працювати 40000 хвилин, відсоток агрегатів, що не дійшли до кінця буде становити 36.7%, а також рекомендовано мати накопичувач з розміром 80. Але якщо регулювальна ділянка буде працювати більше або менше, цей розмір відповідно буде змінюватись. Також ми розрахували завантаження накопичувача, яке становить 36.58%.

Проведемо експеримент, щоб довести, що визначений розмір накопичувача є оптимальним, для цього скористасємось вхідними даними зазначеними в умові задачі. Ми запустимо їх 81 раз, щоб подивитися чи помістяться агрегати у наш накопичувач.

A	B	C	D	E	F	G	H	I	J	K	L	M	N
CreateDelay	FirstRegDelay	SecondRegDelay	FullRegDelay	MaxQ1	MeanQ1	LeftQ1	MaxQ2	MeanQ2	LeftQ2	NumOfSecond	NumOfFull	Fprob	Sload
30	30	30	100	12	0.642657	0	1092	528.7845	1092	1386	385	0.515892	0.184696
30	30	30	100	10	0.599954	0	1074	523.5394	1074	1346	393	0.521507	0.186114
30	30	30	100	10	0.695685	0	1040	498.2367	1040	1350	431	0.521446	0.176617
30	30	30	100	10	0.567131	0	1034	477.759	1031	1312	392	0.520293	0.174683
30	30	30	100	10	0.444554	0	1027	498.286	1025	1290	406	0.52591	0.183126
30	30	30	100	12	0.530804	0	1016	520.2796	1016	1324	423	0.520811	0.188302
30	30	30	100	12	0.733656	0	1012	486.4837	1012	1334	383	0.511176	0.178264
30	30	30	100	12	0.585303	0	1005	501.9321	1005	1326	404	0.515174	0.183522
30	30	30	100	10	0.503671	0	1003	464.6733	1003	1294	382	0.516984	0.17345
30	30	30	100	14	0.890062	4	1001	468.1992	1001	1304	404	0.517877	0.172576
30	30	30	100	16	0.678782	2	999	463.5025	999	1306	384	0.513935	0.172242
30	30	30	100	8	0.526168	0	999	497.8306	999	1322	368	0.508367	0.185136
30	30	30	100	10	0.604825	0	996	484.9455	996	1304	413	0.519351	0.178749
30	30	30	100	12	0.597282	0	991	480.5214	991	1360	402	0.505993	0.174545
30	30	30	100	12	0.559104	0	989	505.1335	989	1374	394	0.501632	0.183219

Рисунок 4.6 – Результати проведення експерименту

Повні результати проведення експерименту наведені в Додатку В. Як можемо побачити по результатам експерименту, лише в 2 випадках з 81 можливих кількість агрегатів які залишились у накопичувачі були більшими за оптимальний розмір. Тому розрахунки виявились вірними.

РОЗДІЛ 5. ІНТЕРПРЕТАЦІЯ РЕЗУЛЬТАТІВ МОДЕЛЮВАННЯ

У дослідженні було проведено перевірку моделі, розраховано приблизний час виконання процесів у системі, побудовано графік залежності ймовірності пропуску первинного регулювання від часу моделювання, визначено оптимальну кількість експериментів та проведено дисперсійний аналіз. Це дозволило краще зрозуміти залежність ймовірності пропуску первинного регулювання від вхідних параметрів системи.

Цільові завдання дослідження – визначити ймовірність відмови в первинному регулюванні та завантаження накопичувача агрегатами, що потребують повного регулювання. Визначити параметри та ввести в систему накопичувач, що забезпечує безвідмовне обслуговування агрегатів, що надходять.

Ймовірність відмови становить приблизно 30-50%. Для її зменшення рекомендовано забирати більше агрегатів на первинне регулювання або приймати менше агрегатів. Для зменшення завантаження на накопичувач рекомендовано пришвидшити час обробки повним регулюванням або забрати більше агрегатів. Відсоток, який залишається у накопичувачі, становить приблизно 30-40%. Завантаження накопичувача агрегатами, які очікують повного регулювання в середньому виходить приблизно 15-20%.

Рекомендований розмір накопичувача для безвідмовної роботи ділянки залежить від часу роботи, оскільки час повного регулювання напряму впливає на результат. При часі роботи в 1440 хвилин, що дорівнює одному дню, рекомендовано поставити накопичувач, який вміщає 40 агрегатів. При часі роботи в 40000 хвилин, цей розмір збільшується до 1060. Для оптимізації розміру необхідно пришвидшити час обробки вторинним та повним регулюванням або додавати додаткові ресурси, тобто обробляти більше агрегатів.

ВИСНОВОК

У цій роботі розглянута проблема визначення оптимального режиму роботи регулювальної ділянки цеху, який забезпечує мінімальну ймовірність того, що обладнання не буде відрегульоване вчасно. Було досліджено можливі методи розв'язання цієї проблеми, розроблено концептуальну та формалізовану моделі, виконано програмну реалізацію моделі та розв'язано поставлену задачу. Також проведено аналіз експериментально отриманих даних.

Для розв'язання поставленої задачі було використано мову програмування Java, бібліотеку PetriObjModelPaint та розроблену імітаційну модель. Простота зміни вхідних даних та результати моделювання дозволяють дослідити роботу регулювальної ділянки.

Відповідно до проведених експериментів, ми можемо сказати, що рекомендований розмір накопичувача за один день дорівнюватиме 30. Також відсоток відмови дорівнює 30-50%, але його можна покращити, якщо пришвидшити первинну обробку, або виділити на неї додаткові ресурси. При тестуванні ми перевірили, що модель є адекватною

.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Основні види моделювання. Формальні методи побудови моделей ПНС ХНЕУ ім. С. Кузнеця – 2017. – 18 с.
2. Імітаційне моделювання [Електронний ресурс] – Режим доступу до ресурсу: [//stud.com.ua/98833/informatika/imitatsiyne_modelyuvannya](https://stud.com.ua/98833/informatika/imitatsiyne_modelyuvannya) // студота інші -педії не наводимо у джерелах, оскільки немає відповідального автора
3. An Introduction to Statistical Learning / Daniela Witten, Gareth M. James, Trevor Hastie, Robert Tibshirani. – 2013 – 440 с.
4. Стеценко І.В. Моделювання систем: Навчальний посібник / І.В. Стеценко; М-во освіти і науки України, Черк. держ. технол. ун-т. – Черкаси: ЧДТУ, 2011. – 407с.
5. Імітаційне моделювання систем та процесів: Електронне навчальне видання. Конспект лекцій / В.Б. Неруш, В.В. Курдеча. – К.: НН ІТС НТУУ «КПІ», 2012. – 115 с.
6. Веб-сервіс моделювання дискретно-подійних систем / Дифучин А.Ю; КПІ ім. Ігоря Сікорського – Київ, 2018. – 95 с.
7. Стеценко І. В. бібліотека «PetriObjModelPaint» - URL: <https://github.com/StetsenkoInna/PetriObjModelPaint>
8. Мова програмування Java [Електронний ресурс] – Режим доступу до ресурсу: <https://www.java.com/en/>

Додаток А. Лістинг коду

```

package LibNet;

import PetriObj.PetriObjModel;
import PetriObj.PetriSim;
import PetriObj.ArcIn;
import PetriObj.ArcOut;
import PetriObj.ExceptionInvalidNetStructure;
import PetriObj.ExceptionInvalidTimeDelay;
import PetriObj.PetriNet;
import PetriObj.PetriP;
import PetriObj.PetriT;
import java.util.ArrayList;
import javax.swing.JFrame;
import javax.swing.JScrollPane;
import javax.swing.JTable;

/**
 *
 * @author white
 */
public class coursework1 {

    public static void main(String[] args) throws
ExceptionInvalidTimeDelay, ExceptionInvalidNetStructure {

        int timeModeling = 1440;

        int numOfExp = 6;

        firstExperiment(timeModeling);
        //showStatistics(numOfExp, timeModeling);
        //validateModel(timeModeling);
        //chebishevExperiment(timeModeling);
        //timeExperiment(numOfExp);

    }

```

```

public static void firstExperiment(int timeModeling)throws
    ExceptionInvalidTimeDelay, ExceptionInvalidNetStructure {
    PetriObjModel model = getModel(30, 30, 1, 30, 1, 100, 1);
    model.setIsProtokol(false);
    model.go(timeModeling);

    double total =
    (double) (model.getListObj().get(0).getNet().getListP()[6].getMark() +
    model.getListObj().get(0).getNet().getListP()[4].getMark())

        + model.getListObj().get(0).getNet().getListP()[3].getMark()) +
    model.getListObj().get(0).getNet().getListP()[2].getMark();

    double failureProbability =
    (model.getListObj().get(0).getNet().getListP()[6].getMark() +
    model.getListObj().get(0).getNet().getListP()[4].getMark()) / total;

    double storageLoad =
    model.getListObj().get(0).getNet().getListP()[4].getMean() / total;

    System.out.println();

    System.out.println("~~~~~Solo
Experiment~~~~~");

    System.out.println("MaxQ1: " +
    model.getListObj().get(0).getNet().getListP()[2].getObservedMax());

    System.out.println("MeanQ1: " +
    model.getListObj().get(0).getNet().getListP()[2].getMean());

    System.out.println("LeftQ1: " +
    model.getListObj().get(0).getNet().getListP()[2].getMark());

    System.out.println("MaxQ2: " +
    model.getListObj().get(0).getNet().getListP()[4].getObservedMax());

    System.out.println("MeanQ2: " +
    model.getListObj().get(0).getNet().getListP()[4].getMean());

    System.out.println("LeftQ2: " +
    model.getListObj().get(0).getNet().getListP()[4].getMark());

    System.out.println("NumOfFull: " +
    model.getListObj().get(0).getNet().getListP()[6].getMark());

    System.out.println("NumOfSecond: " +
    model.getListObj().get(0).getNet().getListP()[3].getMark());

    System.out.println("Failure Prob: " + failureProbability);

    System.out.println("Storage Load: " + storageLoad);

```



```

System.out.println("~~~~~
~");
}

```

```

    public static void showStatistics(int numOfExp, int timeModeling)
throws

```

```

        ExceptionInvalidTimeDelay, ExceptionInvalidNetStructure {

```

```

        JFrame frame = new JFrame();

```

```

        String[] columnNames = {"CreateDelay", "FirstRegDelay",
"SecondRegDelay", "FullRegDelay", "MaxQ1" ,"MeanQ1" ,"LeftQ1" ,"MaxQ2"
,"MeanQ2" ,"LeftQ2", "NumOfSecond" ,"NumOfFull", "FProb", "SLoad"};

```

```

        double[] createDelay = {20.0, 30.0, 40.0};

```

```

        double[] firstRegDelay = {20.0, 30.0, 40.0};

```

```

        double[] secondRegDelay = {20.0, 30.0, 40.0};

```

```

        double[] fullRegDelay = {60.0, 100.0, 140.0};

```

```

        //double[] createDelay = {30.0, 30.0, 30.0};

```

```

        //double[] firstRegDelay = {30.0, 30.0, 30.0};

```

```

        //double[] secondRegDelay = {30.0, 30.0, 30.0};

```

```

        //double[] fullRegDelay = {100.0, 100.0, 100.0};

```

```

        int numoSituations = 81;

```

```

        Object[][] data = new
Object[numoSituations*numOfExp][columnNames.length];

```

```

        int index = 0;

```

```

        ArrayList<Double> probabilities = new ArrayList<>();

```

```

        ArrayList<Double> groupDeviation = new ArrayList<>();

```

```

        ArrayList<Double> probabilitiesAvg = new ArrayList<>();

```

```

        ArrayList<Double> difference = new ArrayList<>();

```

```

        int totalSumOfProcessed = 0;

```

```

        int totalSumOfStucked = 0;

```

```

        int totalSumInStorage = 0;

```

```

double totalSumInStorageMean = 0;

for(int i1 = 0; i1 < createDelay.length; i1++){
    for(int i2 = 0; i2 < firstRegDelay.length; i2++){
        for(int i3 = 0; i3 < secondRegDelay.length; i3++){
            for(int i4 = 0; i4 < fullRegDelay.length; i4++){
                for(int j = 0; j < numOfExp; j++){

                    PetriObjModel model =
getModel(createDelay[i1], firstRegDelay[i2], 1, secondRegDelay[i3], 1,
fullRegDelay[i4], 1);

                    model.setIsProtokol(false);

                    model.go(timeModeling);

                    double total =
((double) (model.getListObj().get(0).getNet().getListP()[6].getMark() +
model.getListObj().get(0).getNet().getListP()[4].getMark())
+
model.getListObj().get(0).getNet().getListP()[3].getMark()) +
model.getListObj().get(0).getNet().getListP()[2].getMark());

                    double failureProbability =
(model.getListObj().get(0).getNet().getListP()[6].getMark() +
model.getListObj().get(0).getNet().getListP()[4].getMark())
/ total;

                    double storageLoad =
model.getListObj().get(0).getNet().getListP()[4].getMean()/total;

                    totalSumOfProcessed = totalSumOfProcessed +
model.getListObj().get(0).getNet().getListP()[3].getMark()+
model.getListObj().get(0).getNet().getListP()[6].getObservedMax();

                    totalSumOfStucked = totalSumOfStucked +
model.getListObj().get(0).getNet().getListP()[2].getMark() +
model.getListObj().get(0).getNet().getListP()[4].getObservedMax();

                    totalSumInStorage = totalSumInStorage +
model.getListObj().get(0).getNet().getListP()[4].getObservedMax();

                    totalSumInStorageMean =
totalSumInStorageMean +
model.getListObj().get(0).getNet().getListP()[4].getMean();

```

```

        data[index] = new Object[]{createDelay[i1],
firstRegDelay[i2], secondRegDelay[i3], fullRegDelay[i4],

model.getListObj().get(0).getNet().getListP()[2].getObservedMax(),

model.getListObj().get(0).getNet().getListP()[2].getMean(),

model.getListObj().get(0).getNet().getListP()[2].getMark(),

model.getListObj().get(0).getNet().getListP()[4].getObservedMax(),

model.getListObj().get(0).getNet().getListP()[4].getMean(),

model.getListObj().get(0).getNet().getListP()[4].getMark(),

model.getListObj().get(0).getNet().getListP()[3].getMark(),

model.getListObj().get(0).getNet().getListP()[6].getMark(),

        failureProbability, storageLoad

        };

        index++;

        probabilities.add(failureProbability);
    }

    double total =
probabilities.stream().mapToDouble(Double::doubleValue).sum();

    double failureProbability = total / numOfExp;
    for(int j = 0; j<numOfExp; j++){
        probabilitiesAvg.add(failureProbability);
    }
}

}

}

}

JTable table = new JTable(data, columnNames);

JScrollPane sp = new JScrollPane(table);

```

```

frame.add(sp);

frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

frame.setSize(1920,1080);

frame.setLocationRelativeTo(null);

frame.setVisible(true);


double total =
probabilities.stream().mapToDouble(Double::doubleValue).sum();

double failureProbability = total / probabilities.size();

double storageLoad = (double) totalSumInStorageMean /
(totalSumOfStucked + totalSumOfProcessed);

double storageAvg = (double) totalSumInStorage /
(numOfSituations * numOfExp);

double stuckProbability = (double) totalSumOfStucked /
(totalSumOfStucked + totalSumOfProcessed);

long adviceForSize = (long) ((Math.round(((totalSumOfStucked +
totalSumOfProcessed)/(numOfSituations * numOfExp))*
stuckProbability)+5)/10)*10;


System.out.println();

System.out.println("~~~~~Main
Goals~~~~~");

System.out.println("Failure Probability: " +
failureProbability);

System.out.println("Storage Load: " + storageLoad);

System.out.println("Storage Avg Max: " + storageAvg);

System.out.println("Stuck Probability : " + stuckProbability);

System.out.println("Advice For Size : " + adviceForSize);


System.out.println("~~~~~
~");

for (int i = 0; i < probabilities.size(); i++) {

    groupDeviation.add(Math.pow((probabilities.get(i) -
probabilitiesAvg.get(i)), 2));

    difference.add(Math.pow((probabilitiesAvg.get(i) -
failureProbability), 2));

}

```

```

        double sFactual =
difference.stream().mapToDouble(Double::doubleValue).sum() * numOfExp;

        double sResidual =
groupDeviation.stream().mapToDouble(Double::doubleValue).sum();

        double degreesOfFreedom = numOfSituations * (numOfExp - 1);
        double dFactual = sFactual;
        double dResidual = sResidual / degreesOfFreedom;
        double f = dFactual / dResidual;
        double fCritical = 1.41805107; //a=0.05; k1=49; k2=200
        System.out.println();

        System.out.println("~~~~~Analysis Of
Variance~~~~~");

        System.out.println("Sum of Squares: " + dFactual);
        System.out.println("Degrees of Freedom: " + degreesOfFreedom);
        System.out.println("Mean Square: " + dResidual);
        System.out.println("F: " + f);
        System.out.println("F-critical (alpha=0.05): " + fCritical);
        if (f > fCritical) {
            System.out.println("Influence is worthy of attention");
        } else {
            System.out.println("Influence is not worthy of attention");
        }

        System.out.println("~~~~~
~");
    }

    public static void validateModel(int timeModeling) throws
ExceptionInvalidTimeDelay, ExceptionInvalidNetStructure {
        PetriObjModel model = getModel(30, 30, 1, 30, 1, 100, 1);
        model.setIsProtokol(false);
        model.go(timeModeling);
        System.out.println();
    }

```

```

        System.out.println("~~~~~Model
Validation~~~~~");

        double practicalT1 = timeModeling /
(double) ((model.getListObj().get(0).getNet().getListP()[3].getMark() +
model.getListObj().get(0).getNet().getListP()[6].getMark() +
model.getListObj().get(0).getNet().getListP()[2].getMark() +
model.getListObj().get(0).getNet().getListP()[4].getMark())/2);

        System.out.println("Theoretical T1 = 30(exp); Factual T1 = " +
practicalT1);

        double practicalT2 = timeModeling /
(double) (model.getListObj().get(0).getNet().getListP()[2].getMark() +
model.getListObj().get(0).getNet().getListP()[3].getMark());

        System.out.println("Theoretical T2 = 30(exp); Factual T2 = "+
practicalT2);

        double practicalT3 = timeModeling /
(double) ((model.getListObj().get(0).getNet().getListP()[3].getMark()));

        System.out.println("Theoretical T3 = 30(exp); Factual T3 = " +
practicalT3);

        double practicalT4 = timeModeling /
(double) ((model.getListObj().get(0).getNet().getListP()[6].getMark()));

        System.out.println("Theoretical T4 = 100(exp); Factual T4 = " +
practicalT4);

System.out.println("~~~~~
~");

    }

    public static void chebishevExperiment(int timeModeling) throws
ExceptionInvalidTimeDelay, ExceptionInvalidNetStructure {

        int runAmount = 100;

        ArrayList<Double> probabilities = new ArrayList<>();

        for (int i = 0; i < runAmount; i++) {

            PetriObjModel model = getModel(30, 30, 1, 30, 1, 100, 1);

            model.setIsProtokol(false);

```

```

        model.go(timeModeling);

        double failureProbability =
(model.getListObj().get(0).getNet().getListP()[6].getMark() +
model.getListObj().get(0).getNet().getListP()[4].getMark())

        /

((double) (model.getListObj().get(0).getNet().getListP()[6].getMark() +
model.getListObj().get(0).getNet().getListP()[4].getMark())

        +
model.getListObj().get(0).getNet().getListP()[3].getMark()) +
model.getListObj().get(0).getNet().getListP()[2].getMark());

        probabilities.add(failureProbability);
    }

    double total =
probabilities.stream().mapToDouble(Double::doubleValue).sum();

    double avg = total / probabilities.size();

    double sum = 0;

    for (Double finProb : probabilities) {
        sum += Math.pow((finProb - avg), 2);
    }

    double stdDev = Math.sqrt(sum / (probabilities.size()-1));

    double numOfExp = Math.pow(stdDev, 2) / (Math.pow((0.05 * avg),
2) * (1 - 0.95));

    System.out.println();

    System.out.println("~~~~~Chebyshev
Test~~~~~");

    System.out.println("Failure Probability = " + avg);

    System.out.println("Standard Deviation = " + stdDev);

    System.out.println("Enough number of experiments by Chebishev: "
+ numOfExp);

System.out.println("~~~~~");
    }

    public static void timeExperiment(int numOfExp) throws
ExceptionInvalidTimeDelay, ExceptionInvalidNetStructure {

        int timeModelingMax = 100000;

```

```

int timeModelingStep = 100;
int timeModelingStart = 100;

JFrame frame = new JFrame();

Object[][] data = new Object[(timeModelingMax-
timeModelingStep)*numOfExp][2];

String[] columnNames = {"CreateDelay",
"FirstRegDelay", "SecondRegDelay"};

int index = 0;
for (int i = 0; i < numOfExp; i++) {
    int t = timeModelingStart;
    while (t <= timeModelingMax) {
        PetriObjModel model = getModel(30, 30, 1, 30, 1, 100,
1);

        model.setIsProtokol(false);

        model.go(t);

        double total =
(double) (model.getListObj().get(0).getNet().getListP()[6].getMark() +
model.getListObj().get(0).getNet().getListP()[4].getMark())
            +
model.getListObj().get(0).getNet().getListP()[3].getMark()) +
model.getListObj().get(0).getNet().getListP()[2].getMark();

        double failureProbability =
(model.getListObj().get(0).getNet().getListP()[6].getMark() +
model.getListObj().get(0).getNet().getListP()[4].getMark())
            / total;

        double storageLoad =
model.getListObj().get(0).getNet().getListP()[4].getMean()/total;

        data[index] = new Object[]{t, failureProbability,
storageLoad};

        t += timeModelingStep;
        index++;
    }
}

```



```

        JTable table = new JTable(data, columnNames);
        JScrollPane sp = new JScrollPane(table);
        frame.add(sp);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(1920,1080);
        frame.setLocationRelativeTo(null);
        frame.setVisible(true);
    }

```

```

    public static PetriObjModel getModel(double createDelay, double
firstRegDelay, int firstRegQ, double secondRegDelay, int secondRegQ,
double fullRegDelay, int fullRegQ) throws ExceptionInvalidTimeDelay,
ExceptionInvalidNetStructure {

```

```

        ArrayList<PetriSim> list = new ArrayList<>();

```

```

        list.add(new PetriSim(CreateAdjSectionNet(createDelay,
firstRegDelay, firstRegQ, secondRegDelay, secondRegQ, fullRegDelay,
fullRegQ)));

```

```

        return new PetriObjModel(list);
    }

```

```

    public static PetriNet CreateAdjSectionNet(double createDelay,
double firstRegDelay, int firstRegQ, double secondRegDelay, int
secondRegQ, double fullRegDelay, int fullRegQ) throws
ExceptionInvalidNetStructure, ExceptionInvalidTimeDelay {

```

```

        ArrayList<PetriP> d_P = new ArrayList<>();

```

```

        ArrayList<PetriT> d_T = new ArrayList<>();

```

```

        ArrayList<ArcIn> d_In = new ArrayList<>();

```

```

        ArrayList<ArcOut> d_Out = new ArrayList<>();

```

```

        d_P.add(new PetriP("Генератор",1));

```

```

        d_P.add(new PetriP("P2",0));

```

```

        d_P.add(new PetriP("Накопичувач",0));

```

```

        d_P.add(new PetriP("Агрегати з вторинного",0));

```

```

        d_P.add(new PetriP("Накопичувач",0));

```

```

        d_P.add(new PetriP("Первинне регулювання вільне", firstRegQ ));

```

```

d_P.add(new PetriP("Агрегати з повного",0));
d_P.add(new PetriP("Вторинне регулювання вільне",secondRegQ));
d_P.add(new PetriP("Повне регулювання вільне",fullRegQ));
d_T.add(new PetriT("Надходження",createDelay));
d_T.get(0).setDistribution("exp", d_T.get(0).getTimeServ());
d_T.get(0).setParamDeviation(0.0);
d_T.add(new PetriT("Первинне регулювання",firstRegDelay));
d_T.get(1).setDistribution("exp", d_T.get(1).getTimeServ());
d_T.get(1).setParamDeviation(0.0);
d_T.get(1).setPriority(10);
d_T.add(new PetriT("Вторинне регулювання",secondRegDelay));
d_T.get(2).setDistribution("exp", d_T.get(2).getTimeServ());
d_T.get(2).setParamDeviation(0.0);
d_T.add(new PetriT("Відмова",0.0));
d_T.add(new PetriT("Повне регулювання", fullRegDelay));
d_T.get(4).setDistribution("exp", d_T.get(4).getTimeServ());
d_T.get(4).setParamDeviation(0.0);
d_In.add(new ArcIn(d_P.get(1),d_T.get(1),2));
d_In.add(new ArcIn(d_P.get(2),d_T.get(2),2));
d_In.add(new ArcIn(d_P.get(1),d_T.get(3),2));
d_In.add(new ArcIn(d_P.get(4),d_T.get(4),1));
d_In.add(new ArcIn(d_P.get(7),d_T.get(2),1));
d_In.add(new ArcIn(d_P.get(8),d_T.get(4),1));
d_In.add(new ArcIn(d_P.get(5),d_T.get(1),1));
d_In.add(new ArcIn(d_P.get(0),d_T.get(0),1));
d_Out.add(new ArcOut(d_T.get(0),d_P.get(1),2));
d_Out.add(new ArcOut(d_T.get(1),d_P.get(2),2));
d_Out.add(new ArcOut(d_T.get(2),d_P.get(3),2));
d_Out.add(new ArcOut(d_T.get(3),d_P.get(4),2));
d_Out.add(new ArcOut(d_T.get(4),d_P.get(6),1));
d_Out.add(new ArcOut(d_T.get(2),d_P.get(7),1));
d_Out.add(new ArcOut(d_T.get(4),d_P.get(8),1));
d_Out.add(new ArcOut(d_T.get(1),d_P.get(5),1));

```

```
d_Out.add(new ArcOut(d_T.get(0),d_P.get(0),1));  
    PetriNet d_Net = new  
PetriNet("coursework_v1.pns",d_P,d_T,d_In,d_Out);  
    PetriP.initNext();  
    PetriT.initNext();  
    ArcIn.initNext();  
    ArcOut.initNext();  
  
    return d_Net;  
}  
}
```

Додаток Б. Результати проведення експериментів

Creat eD	First RegD	Secon d RegD	Full RegD	Q1 Max	Q1 Mean	Q1 Left	Q1 Max	Q1 Mean	Q2 Left	Num Of Secon d	Num OfFull	Fprob	Sload
20	20	20	60	14	0.617 568	0	1419	738.9 033	1419	2070	704	0.506 32	0.176 223
20	20	20	60	12	0.570 558	0	1355	664.7 98	1354	2026	633	0.495 141	0.165 661
20	20	20	60	12	0.590 149	0	1398	714.0 358	1396	2030	653	0.502 329	0.175 052
20	20	20	60	16	0.644 48	0	1241	598.8 579	1241	2092	684	0.479 213	0.149 081
20	20	20	60	10	0.566 498	2	1356	692.3 079	1355	2038	650	0.495 674	0.171 152
20	20	20	60	10	0.600 321	0	1357	658.2 157	1357	2046	640	0.493 94	0.162 804
20	20	20	100	12	0.581 677	0	1688	798.8 629	1688	2034	389	0.505 23	0.194 323
20	20	20	100	14	0.814 193	0	1494	737.5 063	1494	2030	401	0.482 803	0.187 9
20	20	20	100	12	0.627 182	0	1523	735.7 562	1523	2004	394	0.488 906	0.187 645
20	20	20	100	14	0.623 664	2	1586	777.9 309	1586	1994	421	0.501 374	0.194 337
20	20	20	100	12	0.552 798	2	1591	767.0 025	1591	2024	420	0.498 142	0.189 993
20	20	20	100	12	0.821 965	0	1431	763.8 056	1430	1978	437	0.485 566	0.198 649
20	20	20	140	12	0.517 709	0	1823	879.6 084	1823	2008	300	0.513 919	0.212 929
20	20	20	140	12	0.593 449	0	1837	882.1 713	1837	1994	258	0.512 35	0.215 743
20	20	20	140	12	0.619 342	2	1743	869.8 116	1743	2028	294	0.500 861	0.213 871
20	20	20	140	10	0.489 331	0	1785	898.7 792	1784	1952	293	0.515 513	0.223 077
20	20	20	140	10	0.572 955	2	1724	874.3 009	1723	1950	320	0.511 389	0.218 849
20	20	20	140	16	0.687 281	0	1700	892.0 079	1699	1920	254	0.504 26	0.230 314
20	20	30	60	24	2.978 666	2	1242	625.5 936	1242	1950	693	0.497 813	0.160 945
20	20	30	60	22	3.174 827	0	1361	720.1 69	1361	2054	660	0.495 951	0.176 729
20	20	30	60	22	2.935 006	0	1384	692.0 479	1384	2022	713	0.509 104	0.168 014
20	20	30	60	24	3.135 433	0	1361	691.8 08	1358	2018	665	0.500 619	0.171 197
20	20	30	60	26	3.780 32	0	1322	711.2 893	1321	1998	664	0.498 368	0.178 581
20	20	30	60	28	3.355 833	0	1407	692.7 931	1403	2022	660	0.505 018	0.169 594

20	20	30	100	18	2.299 375	0	1491	758.0 2	1491	1998	386	0.484 387	0.195 618
20	20	30	100	28	3.714 095	0	1548	763.3 747	1548	2052	377	0.484 033	0.191 947
20	20	30	100	32	2.423 596	4	1650	793.3 447	1650	1910	407	0.518 006	0.199 785
20	20	30	100	28	5.389 828	0	1582	788.9 155	1582	2054	395	0.490 449	0.195 712
20	20	30	100	18	2.042 394	0	1659	853.2 949	1658	1940	395	0.514 15	0.213 698
20	20	30	100	26	3.386 133	0	1602	842.6 99	1601	2074	414	0.492 786	0.206 089
20	20	30	140	22	2.756 214	6	1635	834.7 452	1635	1918	320	0.503 996	0.215 196
20	20	30	140	32	2.866 616	0	1780	877.9 752	1779	1958	274	0.511 842	0.218 892
20	20	30	140	42	4.417 794	0	1750	878.7 479	1750	1968	263	0.505 652	0.220 735
20	20	30	140	30	3.763 854	0	1679	873.4 44	1679	2008	292	0.495 351	0.219 513
20	20	30	140	24	3.588 342	0	1715	866.9 136	1714	2018	303	0.499 876	0.214 848
20	20	30	140	20	2.900 091	2	1698	811.0 922	1698	1920	275	0.506 547	0.208 239
20	20	40	60	84	35.89 666	68	1176	555.7 023	1175	1990	684	0.474 598	0.141 869
20	20	40	60	66	23.49 116	4	1301	668.1 301	1300	1966	689	0.502 4	0.168 762
20	20	40	60	154	61.53 512	58	1219	585.3 886	1219	1974	630	0.476 424	0.150 834
20	20	40	60	156	77.58 457	156	1257	648.9 859	1257	1956	646	0.473 973	0.161 64
20	20	40	60	82	21.42 443	28	1403	700.4 07	1401	1976	666	0.507 738	0.172 048
20	20	40	60	102	38.67 341	26	1304	646.4 883	1304	1958	657	0.497 085	0.163 875
20	20	40	100	138	53.11 113	132	1605	792.3 402	1603	1938	400	0.491 775	0.194 535
20	20	40	100	58	24.17 195	6	1688	818.9 748	1688	2012	421	0.511 025	0.198 443
20	20	40	100	80	41.50 782	24	1478	766.3 66	1478	1942	401	0.488 687	0.199 315
20	20	40	100	96	47.65 862	58	1361	697.9 083	1361	1972	430	0.468 725	0.182 651
20	20	40	100	108	40.53 663	50	1667	821.9 075	1666	1928	409	0.511 966	0.202 79
20	20	40	100	66	26.93 779	14	1502	750.2 102	1500	1968	407	0.490 357	0.192 906
20	20	40	140	100	60.42 372	92	1828	930.8 737	1825	1862	292	0.520 02	0.228 66
20	20	40	140	50	22.54 361	14	1549	754.0 361	1548	2024	315	0.477 57	0.193 293
20	20	40	140	42	14.34 701	20	1730	873.7 929	1729	1930	288	0.508 445	0.220 265
20	20	40	140	54	23.34 844	52	1684	825.5 07	1684	1942	291	0.497 606	0.207 989

20	20	40	140	214	102.3 668	208	1765	866.8 472	1764	1816	265	0.500 617	0.213 878
20	20	40	140	58	14.98 583	22	1634	766.3 378	1632	1930	283	0.495 216	0.198 174
20	30	20	60	8	0.326 961	0	1861	938.1 401	1860	1514	647	0.623 477	0.233 31
20	30	20	60	8	0.239 302	0	1670	868.8 33	1668	1592	663	0.594 188	0.221 472
20	30	20	60	8	0.276 435	0	1644	841.2 372	1641	1600	644	0.588 16	0.216 535
20	30	20	60	10	0.277 945	0	1704	849.2 346	1704	1552	635	0.601 131	0.218 256
20	30	20	60	6	0.172 878	0	1802	889.7 215	1801	1544	672	0.615 634	0.221 489
20	30	20	60	10	0.386 62	0	1544	777.0 447	1544	1618	657	0.576 329	0.203 468
20	30	20	100	10	0.384 366	0	2014	1012. 474	2013	1568	386	0.604 739	0.255 224
20	30	20	100	8	0.271 869	0	1983	976.7 466	1983	1544	388	0.605 619	0.249 488
20	30	20	100	6	0.205 697	0	1979	1035. 241	1978	1518	381	0.608 46	0.267 021
20	30	20	100	6	0.325 895	0	1894	958.8 271	1892	1598	429	0.592 243	0.244 661
20	30	20	100	8	0.338 507	0	1994	988.2 232	1994	1634	383	0.592 62	0.246 378
20	30	20	100	10	0.324 831	0	1974	966.0 112	1974	1590	417	0.600 603	0.242 655
20	30	20	140	14	0.266 922	2	2123	1074. 481	2123	1586	260	0.600 101	0.270 582
20	30	20	140	10	0.287 829	0	2285	1151. 069	2285	1558	302	0.624 125	0.277 701
20	30	20	140	6	0.229 926	0	2014	1015. 744	2014	1584	301	0.593 742	0.260 514
20	30	20	140	12	0.373 754	0	2069	1034. 033	2069	1584	324	0.601 71	0.260 003
20	30	20	140	10	0.230 022	0	2220	1086. 507	2218	1492	283	0.626 346	0.272 103
20	30	20	140	6	0.195 153	0	2334	1153. 005	2334	1594	279	0.621 108	0.274 068
20	30	30	60	10	0.908 182	6	1558	763.9 885	1558	1584	737	0.590 734	0.196 651
20	30	30	60	20	2.030 877	0	1872	935.4 443	1870	1632	637	0.605 702	0.226 007
20	30	30	60	20	1.457 402	8	1714	853.5 298	1713	1616	666	0.594 304	0.213 223
20	30	30	60	18	1.000 589	0	1774	843.9 676	1774	1592	685	0.607 011	0.208 336
20	30	30	60	10	0.978 327	0	1737	858.0 63	1736	1512	643	0.611 411	0.220 525
20	30	30	60	18	1.099 992	0	1636	800.9 461	1636	1604	657	0.588 401	0.205 529
20	30	30	100	12	0.923 337	0	2035	1102. 266	2035	1620	364	0.596 915	0.274 264
20	30	30	100	16	0.971 458	0	1890	961.2 238	1890	1560	409	0.595 75	0.249 086

20	30	30	100	16	0.933 925	0	2067	1038. 423	2067	1558	414	0.614 261	0.257 099
20	30	30	100	16	1.467 015	4	2189	1092. 544	2189	1592	396	0.618 273	0.261 312
20	30	30	100	14	0.830 571	0	2102	1053. 637	2100	1548	409	0.618 437	0.259 708
20	30	30	100	14	1.148 487	0	2002	1002. 001	2002	1552	399	0.607 387	0.253 479
20	30	30	140	22	1.275 551	0	2013	974.7 745	2012	1570	317	0.597 333	0.250 006
20	30	30	140	12	1.565 351	4	2109	1033. 015	2109	1666	294	0.589 983	0.253 625
20	30	30	140	12	1.371 318	0	2230	1151. 364	2230	1660	259	0.599 904	0.277 504
20	30	30	140	22	1.482 092	0	2037	1016. 994	2037	1644	318	0.588 897	0.254 312
20	30	30	140	18	1.199 664	0	2134	1094. 42	2134	1560	285	0.607 942	0.275 049
20	30	30	140	18	1.862 164	10	2201	1091. 605	2201	1606	288	0.606 334	0.265 921
20	30	40	60	32	5.738 383	0	1731	869.3 304	1730	1568	671	0.604 938	0.219 03
20	30	40	60	16	2.701 485	2	1499	765.7 546	1496	1600	695	0.577 643	0.201 886
20	30	40	60	14	2.609 237	6	1788	896.3 759	1787	1556	690	0.613 271	0.221 93
20	30	40	60	22	3.500 977	0	1787	871.7 21	1782	1640	665	0.598 728	0.213 291
20	30	40	60	30	4.545 813	0	1653	826.4 945	1651	1598	678	0.593 074	0.210 465
20	30	40	60	24	4.190 533	0	1642	831.6 883	1640	1638	709	0.589 165	0.208 6
20	30	40	100	32	3.614 419	0	1960	920.1 986	1960	1518	415	0.610 069	0.236 373
20	30	40	100	26	3.442 976	0	1934	1004. 085	1934	1656	381	0.582 977	0.252 854
20	30	40	100	22	3.548 3	10	1909	957.6 07	1909	1564	416	0.596 307	0.245 603
20	30	40	100	26	3.590 109	0	1808	916.7 982	1808	1616	407	0.578 178	0.239 31
20	30	40	100	22	3.397 389	0	1945	971.4 672	1945	1588	410	0.597 261	0.246 378
20	30	40	100	26	4.532 07	0	1959	966.4 07	1958	1532	401	0.606 271	0.248 37
20	30	40	140	16	2.709 624	2	2165	1113. 617	2165	1566	286	0.609 853	0.277 088
20	30	40	140	22	2.903 122	6	1974	991.5 365	1974	1542	271	0.591 88	0.261 412
20	30	40	140	22	3.966 3	8	2246	1125. 532	2246	1506	289	0.626 081	0.277 978
20	30	40	140	20	3.669 377	16	2186	1060. 771	2186	1604	315	0.606 892	0.257 406
20	30	40	140	26	5.328 471	12	2130	1023. 852	2130	1646	283	0.592 729	0.251 499
20	30	40	140	28	3.112 326	6	2136	1090. 561	2133	1552	290	0.608 641	0.273 941

20	40	20	60	6	0.152 343	0	1877	928.8 874	1877	1358	668	0.652 063	0.237 993
20	40	20	60	6	0.142 185	0	2025	972.4 044	2025	1312	652	0.671 096	0.243 771
20	40	20	60	6	0.171 709	2	1981	1023. 327	1980	1320	697	0.669 417	0.255 896
20	40	20	60	8	0.174 966	2	2067	1032. 901	2067	1304	682	0.677 928	0.254 723
20	40	20	60	6	0.139 831	4	2009	969.8 765	2009	1286	680	0.675 798	0.243 749
20	40	20	60	4	0.121 65	0	2044	996.3 394	2044	1360	631	0.662 949	0.246 924
20	40	20	100	8	0.174 562	0	2264	1107. 835	2264	1308	437	0.673 734	0.276 337
20	40	20	100	6	0.129 048	0	2327	1178. 73	2327	1402	406	0.660 943	0.285 062
20	40	20	100	8	0.152 838	0	2310	1153. 734	2308	1342	389	0.667 74	0.285 648
20	40	20	100	6	0.169 927	0	2414	1195. 674	2414	1356	399	0.674 742	0.286 801
20	40	20	100	8	0.179 521	0	2330	1179. 262	2330	1388	417	0.664 329	0.285 19
20	40	20	100	6	0.174 577	0	2266	1123. 789	2266	1374	385	0.658 634	0.279 202
20	40	20	140	10	0.251 997	0	2340	1151. 566	2340	1374	275	0.655 553	0.288 685
20	40	20	140	6	0.132 25	0	2390	1224. 191	2390	1292	259	0.672 164	0.310 629
20	40	20	140	8	0.196 994	0	2347	1129. 951	2346	1340	281	0.662 213	0.284 838
20	40	20	140	6	0.148 145	0	2330	1175. 898	2330	1350	277	0.658 832	0.297 169
20	40	20	140	8	0.224 933	0	2410	1125. 231	2409	1388	260	0.657 875	0.277 355
20	40	20	140	8	0.179 711	0	2396	1168. 737	2395	1396	290	0.657 927	0.286 385
20	40	30	60	12	0.685 773	0	1969	946.5 7	1969	1360	668	0.659 745	0.236 82
20	40	30	60	12	0.540 934	0	2128	1043. 773	2128	1290	653	0.683 125	0.256 392
20	40	30	60	14	0.603 277	6	1970	975.0 461	1970	1348	629	0.657 475	0.246 66
20	40	30	60	16	0.402 482	0	1921	992.1 292	1921	1304	694	0.667 262	0.253 159
20	40	30	60	16	0.853 772	0	2052	1005. 559	2050	1380	601	0.657 653	0.249 456
20	40	30	60	12	0.627 681	4	1838	916.0 351	1836	1346	687	0.651 433	0.236 518
20	40	30	100	16	0.707 954	0	2238	1162. 9	2238	1288	399	0.671 847	0.296 28
20	40	30	100	10	0.568 211	0	2302	1136. 317	2302	1318	373	0.669 922	0.284 577
20	40	30	100	8	0.675 821	0	2197	1085. 663	2197	1352	408	0.658 327	0.274 365
20	40	30	100	14	0.737 081	0	2317	1142. 462	2316	1338	385	0.668 73	0.282 858

20	40	30	100	16	0.651 295	2	2187	1104. 539	2187	1336	394	0.658 586	0.281 842
20	40	30	100	10	0.653 281	0	2179	1095. 608	2179	1360	438	0.658 034	0.275 486
20	40	30	140	10	0.470 619	0	2457	1201. 384	2457	1252	282	0.686 294	0.301 023
20	40	30	140	10	0.497 931	0	2494	1272. 581	2494	1286	293	0.684 262	0.312 443
20	40	30	140	10	0.562 65	0	2434	1254. 616	2434	1376	287	0.664 144	0.306 228
20	40	30	140	14	0.526 249	0	2371	1168. 474	2371	1308	300	0.671 274	0.293 66
20	40	30	140	22	0.706 567	0	2316	1147. 76	2315	1284	280	0.668 987	0.295 891
20	40	30	140	10	0.688 253	0	2282	1119. 754	2282	1376	297	0.652 086	0.283 124
20	40	40	60	16	1.442 887	0	2076	1032. 875	2074	1358	669	0.668 861	0.251 859
20	40	40	60	24	2.050 878	0	1987	965.3 419	1986	1330	667	0.666 081	0.242 366
20	40	40	60	20	1.478 184	0	1981	972.5 77	1979	1324	680	0.667 587	0.244 182
20	40	40	60	16	1.604 921	0	2026	1022. 244	2026	1308	663	0.672 755	0.255 753
20	40	40	60	24	2.127 792	0	2072	1046. 535	2072	1354	671	0.669 514	0.255 439
20	40	40	60	18	1.957 385	0	2110	1029. 015	2110	1312	687	0.680 701	0.250 43
20	40	40	100	12	1.672 632	8	2177	1098. 713	2177	1314	404	0.661 286	0.281 505
20	40	40	100	16	1.070 446	0	2288	1162. 784	2288	1352	369	0.662 759	0.290 043
20	40	40	100	16	1.915 177	0	2280	1125. 201	2279	1348	398	0.665 093	0.279 553
20	40	40	100	16	1.923 752	8	2243	1132. 277	2242	1324	401	0.664 906	0.284 849
20	40	40	100	16	1.487 667	2	2176	1123. 259	2176	1304	407	0.664 181	0.288 83
20	40	40	100	18	1.663 469	2	2248	1125. 187	2248	1302	429	0.672 444	0.282 639
20	40	40	140	14	1.685 969	0	2442	1186. 62	2441	1344	286	0.669 86	0.291 481
20	40	40	140	14	1.606 964	6	2513	1257. 486	2513	1312	278	0.679 241	0.306 032
20	40	40	140	22	1.737 155	0	2452	1223. 869	2452	1286	297	0.681 289	0.303 313
20	40	40	140	12	1.499 352	0	2400	1231. 244	2400	1314	271	0.670 263	0.308 97
20	40	40	140	20	2.408 315	0	2207	1060. 171	2207	1354	302	0.649 495	0.274 442
20	40	40	140	12	1.384 091	0	2292	1162. 473	2291	1356	290	0.655 575	0.295 269
30	20	20	60	12	0.440 923	0	386	205.1 724	385	1716	726	0.392 996	0.072 576
30	20	20	60	8	0.210 929	0	530	257.9 33	528	1548	611	0.423 893	0.095 993

30	20	20	60	8	0.269 121	0	410	241.2 785	389	1536	686	0.411 72	0.092 408
30	20	20	60	12	0.297 167	0	475	229.5 047	467	1590	630	0.408 262	0.085 413
30	20	20	60	6	0.202 739	0	355	186.3 908	351	1530	666	0.399 293	0.073 181
30	20	20	60	8	0.232 771	0	312	169.3 015	308	1556	653	0.381 804	0.067 263
30	20	20	100	8	0.409 248	0	696	328.2 243	696	1608	407	0.406 861	0.121 071
30	20	20	100	8	0.305 827	0	778	357.8 154	778	1634	381	0.414 966	0.128 112
30	20	20	100	8	0.273 765	0	663	333.0 488	662	1620	411	0.398 44	0.123 672
30	20	20	100	6	0.281 142	4	598	292.9 025	596	1540	383	0.388 03	0.116 093
30	20	20	100	8	0.288 998	0	644	312.7 323	644	1626	365	0.382 922	0.118 684
30	20	20	100	8	0.354 419	0	790	396.5 885	790	1614	367	0.417 539	0.143 121
30	20	20	140	12	0.319 992	0	756	366.0 255	755	1614	282	0.391 173	0.138 071
30	20	20	140	8	0.248 815	0	732	365.2 728	732	1578	257	0.385 275	0.142 296
30	20	20	140	12	0.315 878	0	792	399.0 79	791	1574	282	0.405 365	0.150 767
30	20	20	140	10	0.341 904	0	786	365.5 554	786	1696	283	0.386 618	0.132 208
30	20	20	140	8	0.303 78	0	742	364.6 349	742	1616	299	0.391 795	0.137 236
30	20	20	140	8	0.294 841	4	778	370.1 28	778	1610	257	0.390 713	0.139 724
30	20	30	60	12	1.239 836	0	419	246.7 178	413	1554	644	0.404 826	0.094 492
30	20	30	60	20	1.691 095	4	463	224.7 069	461	1618	654	0.407 38	0.082 1
30	20	30	60	14	1.137 532	6	498	250.5 404	495	1624	664	0.415 561	0.089 832
30	20	30	60	28	1.469 471	0	479	247.6 509	478	1612	641	0.409 74	0.090 681
30	20	30	60	18	1.778 06	4	472	240.9 715	471	1678	644	0.398 641	0.086 154
30	20	30	60	16	1.285 485	0	425	217.1 048	408	1642	665	0.395 212	0.079 965
30	20	30	100	16	0.923 213	0	715	355.6 606	706	1528	455	0.431 759	0.132 265
30	20	30	100	14	1.373 861	0	722	373.9 901	721	1638	354	0.396 24	0.137 851
30	20	30	100	12	1.158 19	0	544	247.3 037	544	1592	393	0.370 502	0.097 787
30	20	30	100	12	1.046 875	4	711	351.6 261	711	1586	406	0.412 634	0.129 895
30	20	30	100	16	0.939 803	0	697	325.3 92	694	1576	409	0.411 721	0.121 46
30	20	30	100	12	1.017 835	2	715	357.3 112	714	1556	387	0.414 065	0.134 378

30	20	30	140	12	0.943 238	0	748	353.3 097	748	1568	303	0.401 298	0.134 903
30	20	30	140	16	0.924 919	6	828	428.9 635	827	1526	304	0.424 709	0.161 083
30	20	30	140	12	1.054 246	0	795	393.2 738	793	1578	300	0.409 21	0.147 238
30	20	30	140	18	1.620 293	0	662	312.8 352	662	1640	283	0.365 571	0.121 019
30	20	30	140	18	1.245 618	0	708	351.5 043	708	1608	329	0.392 06	0.132 894
30	20	30	140	14	1.075 742	2	747	377.4 651	744	1560	291	0.398 537	0.145 347
30	20	40	60	34	6.004 802	0	445	195.1 273	444	1598	647	0.405 727	0.072 565
30	20	40	60	20	2.862 467	12	438	200.3 231	437	1504	642	0.415 8	0.077 196
30	20	40	60	20	2.757 42	0	331	130.3 662	316	1530	663	0.390 195	0.051 959
30	20	40	60	30	5.992 793	8	374	160.0 172	370	1644	681	0.388 827	0.059 2
30	20	40	60	30	3.932 779	0	392	191.3 365	392	1544	657	0.404 551	0.073 79
30	20	40	60	22	3.353 006	2	566	318.5 468	561	1696	674	0.421 071	0.108 608
30	20	40	100	36	5.444 849	8	646	330.0 604	645	1556	402	0.400 996	0.126 412
30	20	40	100	24	4.082 441	6	694	335.1 012	693	1646	368	0.391 08	0.123 517
30	20	40	100	60	12.29 918	0	686	356.0 592	686	1616	407	0.403 47	0.131 436
30	20	40	100	20	3.111 111	0	674	322.1 436	671	1552	372	0.401 927	0.124 14
30	20	40	100	28	4.702 771	0	757	371.3 748	757	1632	388	0.412 315	0.133 732
30	20	40	100	30	4.021 539	4	754	343.3 44	754	1590	423	0.424 756	0.123 906
30	20	40	140	18	3.121 817	0	699	354.6 47	699	1582	254	0.375 937	0.139 9
30	20	40	140	28	4.927 578	0	738	364.2 847	738	1696	285	0.376 241	0.133 977
30	20	40	140	22	3.555 9	0	783	377.3 507	779	1584	274	0.399 317	0.143 098
30	20	40	140	30	6.338 139	4	736	352.8 972	730	1668	271	0.374 486	0.132 023
30	20	40	140	24	3.036 829	14	775	369.8 235	775	1648	300	0.392 766	0.135 12
30	20	40	140	24	4.116 645	0	790	378.2 604	790	1656	285	0.393 629	0.138 506
30	30	20	60	6	0.175 888	0	687	326.2 173	683	1286	678	0.514 167	0.123 24
30	30	20	60	6	0.129 577	6	730	378.4 017	718	1278	659	0.517 475	0.142 203
30	30	20	60	6	0.159 169	0	665	332.6 853	665	1316	692	0.507 669	0.124 461
30	30	20	60	6	0.218 259	0	644	311.1 521	644	1420	637	0.474 269	0.115 199

30	30	20	60	6	0.202 056	0	689	330.6 771	688	1386	653	0.491 749	0.121 26
30	30	20	60	6	0.147 511	0	630	319.7 518	627	1258	688	0.511 077	0.124 272
30	30	20	100	6	0.150 014	0	913	456.8 631	912	1316	379	0.495 205	0.175 245
30	30	20	100	6	0.170 808	0	922	478.2 825	920	1376	407	0.490 936	0.176 945
30	30	20	100	6	0.217 62	0	996	498.1 012	996	1366	381	0.502 005	0.181 59
30	30	20	100	6	0.144 29	0	941	425.9 549	941	1318	414	0.506 921	0.159 355
30	30	20	100	6	0.157 001	0	905	435.9 145	901	1306	404	0.499 809	0.166 953
30	30	20	100	8	0.170 203	0	890	454.8 684	890	1306	397	0.496 336	0.175 422
30	30	20	140	8	0.133 644	0	967	448.9 264	967	1328	288	0.485 869	0.173 8
30	30	20	140	8	0.152 147	0	990	467.3 765	990	1328	251	0.483 067	0.181 929
30	30	20	140	6	0.139 804	0	1147	545.2 991	1147	1372	272	0.508 42	0.195 378
30	30	20	140	8	0.187 603	2	1033	513.7 958	1033	1352	272	0.490 786	0.193 229
30	30	20	140	6	0.189 372	0	1075	521.0 337	1073	1368	280	0.497 244	0.191 486
30	30	20	140	8	0.242 6	0	1146	567.6 56	1146	1384	313	0.513 19	0.199 668
30	30	30	60	14	0.722 863	0	608	318.5 603	607	1352	676	0.486 907	0.120 896
30	30	30	60	10	0.716 814	0	818	442.6 981	814	1374	625	0.511 554	0.157 376
30	30	30	60	12	0.582 997	6	719	342.8 057	719	1286	648	0.514 103	0.128 923
30	30	30	60	8	0.573 563	0	607	318.9 569	605	1310	642	0.487 681	0.124 739
30	30	30	60	10	0.465 262	0	705	347.9 631	703	1252	712	0.530 559	0.130 47
30	30	30	60	12	0.551 545	0	616	310.0 449	616	1366	687	0.488 198	0.116 165
30	30	30	100	14	0.610 112	0	936	461.0 316	936	1360	403	0.496 11	0.170 816
30	30	30	100	10	0.487 773	0	999	522.4 737	999	1330	382	0.509 406	0.192 724
30	30	30	100	10	0.672 639	0	1067	548.3 141	1066	1364	385	0.515 453	0.194 783
30	30	30	100	10	0.611 379	0	858	445.2 474	857	1372	414	0.480 893	0.168 463
30	30	30	100	16	0.888 586	0	954	475.5 311	950	1360	415	0.500 917	0.174 507
30	30	30	100	8	0.469 526	0	1006	477.1 676	1006	1354	359	0.502 023	0.175 494
30	30	30	140	14	0.771 936	2	1016	529.7 947	1015	1352	264	0.485 758	0.201 213
30	30	30	140	18	0.818 8	2	975	522.3 867	975	1402	278	0.471 584	0.196 608

30	30	30	140	8	0.552 77	0	1085	554.0 142	1085	1318	286	0.509 855	0.206 03
30	30	30	140	14	0.709 987	0	1015	508.2 141	1013	1298	306	0.504 012	0.194 197
30	30	30	140	14	0.807 134	0	1087	535.3 605	1087	1346	278	0.503 504	0.197 477
30	30	30	140	10	0.637 384	0	983	501.5 974	983	1364	340	0.492 371	0.186 676
30	30	40	60	14	1.609 309	0	704	353.8 672	699	1350	646	0.499 072	0.131 305
30	30	40	60	16	1.485 606	6	697	307.2 758	697	1304	620	0.501 332	0.116 968
30	30	40	60	18	2.037 341	4	815	400.6 233	814	1342	645	0.520 143	0.142 825
30	30	40	60	16	1.628 119	2	590	298.2 233	589	1354	682	0.483 822	0.113 522
30	30	40	60	18	1.412 119	2	644	328.6 993	644	1308	701	0.506 591	0.123 804
30	30	40	60	22	2.087 139	0	609	300.0 058	605	1308	644	0.488 463	0.117 327
30	30	40	100	20	2.401 737	0	927	499.1 556	927	1330	422	0.503 546	0.186 322
30	30	40	100	18	1.852 766	0	1013	469.7 924	1013	1340	366	0.507 172	0.172 781
30	30	40	100	12	1.309 418	2	1016	445.6 433	1015	1330	390	0.513 336	0.162 822
30	30	40	100	14	1.390 27	0	801	398.0 15	798	1404	447	0.469 989	0.150 251
30	30	40	100	16	1.466 666	0	972	470.1 455	972	1314	399	0.510 615	0.175 101
30	30	40	100	14	1.106 096	0	975	481.4 869	973	1316	392	0.509 138	0.179 592
30	30	40	140	14	1.235 564	0	1031	515.5 641	1030	1266	289	0.510 251	0.199 445
30	30	40	140	16	2.119 345	4	1091	510.7 241	1091	1382	272	0.495 817	0.185 785
30	30	40	140	22	2.546 952	6	1143	564.0 203	1142	1322	307	0.521 786	0.203 104
30	30	40	140	22	2.258 069	10	1074	537.3 636	1074	1346	289	0.501 287	0.197 633
30	30	40	140	20	1.631 034	0	1058	539.5 224	1058	1330	265	0.498 681	0.203 363
30	30	40	140	16	1.701 92	2	871	416.3 395	871	1342	302	0.466 031	0.165 411
30	40	20	60	6	0.089 818	0	743	394.8 283	740	1046	669	0.573 931	0.160 826
30	40	20	60	6	0.094 101	0	937	477.4 198	936	1124	629	0.582 001	0.177 545
30	40	20	60	8	0.099 375	6	917	461.0 779	917	1138	614	0.572 336	0.172 366
30	40	20	60	4	0.038 528	0	965	494.5 168	962	1068	675	0.605 176	0.182 816
30	40	20	60	8	0.137 193	0	898	414.7 988	893	1136	680	0.580 657	0.153 119
30	40	20	60	6	0.123 305	0	937	460.5 878	937	1158	654	0.578 756	0.167 547

30	40	20	100	8	0.120 058	0	1140	580.2 724	1138	1126	387	0.575 255	0.218 888
30	40	20	100	6	0.145 907	0	1040	506.6 916	1038	1154	397	0.554 268	0.195 709
30	40	20	100	8	0.134 774	0	1115	569.7 023	1115	1224	420	0.556 361	0.206 489
30	40	20	100	6	0.081 062	0	1066	528.7 048	1066	1124	411	0.567 859	0.203 27
30	40	20	100	10	0.176 814	0	1017	538.6 898	1013	1176	368	0.540 086	0.210 673
30	40	20	100	6	0.106 339	0	1132	590.8 951	1131	1120	394	0.576 56	0.223 401
30	40	20	140	6	0.084 987	0	1265	615.9 448	1264	1148	283	0.574 026	0.228 551
30	40	20	140	6	0.120 586	0	1202	605.4 657	1202	1170	285	0.559 654	0.227 876
30	40	20	140	6	0.063 08	0	1292	645.0 502	1292	1124	285	0.583 858	0.238 819
30	40	20	140	6	0.105 925	0	1220	585.3 538	1220	1158	289	0.565 804	0.219 48
30	40	20	140	6	0.103 03	0	1257	602.9 127	1257	1200	276	0.560 922	0.220 605
30	40	20	140	6	0.125 42	0	1129	582.7 99	1128	1152	285	0.550 877	0.227 212
30	40	30	60	8	0.284 083	2	996	487.9 302	994	1072	693	0.611 011	0.176 722
30	40	30	60	8	0.285 681	0	775	407.8 386	771	1080	684	0.573 964	0.160 883
30	40	30	60	10	0.414 007	0	884	468.3 919	883	1162	688	0.574 826	0.171 384
30	40	30	60	10	0.474 455	2	720	399.0 725	719	1172	666	0.541 227	0.155 949
30	40	30	60	8	0.368 63	0	852	408.8 251	844	1156	687	0.569 78	0.152 149
30	40	30	60	12	0.437 388	0	808	431.5 262	805	1116	702	0.574 533	0.164 516
30	40	30	100	10	0.408 519	0	1086	533.8 505	1086	1132	421	0.571 05	0.202 293
30	40	30	100	10	0.462 436	0	1246	625.0 191	1246	1140	387	0.588 893	0.225 395
30	40	30	100	14	0.445 408	0	1133	601.8 128	1132	1126	397	0.575 895	0.226 672
30	40	30	100	14	0.406 473	0	1288	616.0 611	1288	1166	403	0.591 88	0.215 632
30	40	30	100	10	0.395 167	0	1226	606.9 028	1226	1052	377	0.603 766	0.228 589
30	40	30	100	8	0.233 174	0	1034	514.5 479	1033	1156	410	0.555 214	0.197 979
30	40	30	140	6	0.317 142	0	1284	648.6 589	1284	1108	269	0.583 615	0.243 765
30	40	30	140	10	0.375 248	0	1255	643.2 229	1255	1128	292	0.578 318	0.240 457
30	40	30	140	8	0.376 265	0	1153	574.4 923	1153	1150	282	0.555 126	0.222 241
30	40	30	140	8	0.354 587	0	1154	581.5 519	1153	1156	320	0.560 289	0.221 207

30	40	30	140	10	0.286 204	0	1350	671.3 216	1350	1116	249	0.588 95	0.247 264
30	40	30	140	10	0.435 308	0	1245	623.9 809	1245	1158	296	0.570 952	0.231 19
30	40	40	60	12	1.032 357	0	999	530.5 039	996	1180	613	0.576 909	0.190 213
30	40	40	60	12	0.681 265	0	795	409.8 856	791	1150	678	0.560 901	0.156 505
30	40	40	60	12	1.078 38	0	978	477.9 157	978	1168	651	0.582 41	0.170 867
30	40	40	60	14	1.007 854	0	891	468.6 135	891	1084	682	0.592 021	0.176 369
30	40	40	60	10	0.985 928	0	873	403.3 42	871	1170	666	0.567 787	0.149
30	40	40	60	16	1.288 504	4	783	373.4 632	782	1190	639	0.543 403	0.142 816
30	40	40	100	10	1.061 357	0	1288	653.5 421	1287	1166	378	0.588 131	0.230 852
30	40	40	100	12	1.144 295	0	1066	525.3 263	1060	1144	393	0.559 492	0.202 282
30	40	40	100	10	0.880 332	0	1017	532.9 926	1017	1120	424	0.562 671	0.208 119
30	40	40	100	18	1.455 473	0	1165	559.0 647	1165	1212	362	0.557 503	0.204 113
30	40	40	100	10	0.682 272	0	1129	566.6 918	1127	1120	386	0.574 63	0.215 227
30	40	40	100	10	0.777 194	0	1048	567.3 292	1048	1130	381	0.558 421	0.221 7
30	40	40	140	16	1.257 188	6	1401	714.4 337	1397	1136	280	0.594 892	0.253 435
30	40	40	140	16	1.097 189	0	1270	651.2 064	1269	1116	280	0.581 238	0.244 355
30	40	40	140	10	0.720 821	0	1197	599.5 817	1196	1122	277	0.567 63	0.231 053
30	40	40	140	14	1.149 14	0	1261	596.9 536	1261	1158	292	0.572 851	0.220 197
30	40	40	140	16	1.309 163	0	1268	651.0 966	1268	1130	269	0.576 303	0.244 131
30	40	40	140	16	1.182 082	2	1186	596.3 573	1186	1132	283	0.564 349	0.229 104
40	20	20	60	10	0.250 052	0	91	40.38 991	60	1352	627	0.336 93	0.019 809
40	20	20	60	6	0.209 928	0	33	13.48 331	7	1286	624	0.329 16	0.007 034
40	20	20	60	8	0.153 524	0	37	10.76 398	24	1254	625	0.341 04	0.005 656
40	20	20	60	6	0.149 421	0	73	43.33 318	0	1332	655	0.329 643	0.021 808
40	20	20	60	6	0.141 309	0	89	36.77 889	87	1262	650	0.368 684	0.018 399
40	20	20	60	4	0.129 869	0	45	15.25 022	16	1310	655	0.338 718	0.007 698
40	20	20	100	8	0.134 461	0	296	141.6 791	291	1338	400	0.340 562	0.069 827
40	20	20	100	6	0.197 515	0	169	86.91 142	168	1340	403	0.298 796	0.045 48

40	20	20	100	6	0.235 88	0	318	156.5 45	315	1366	408	0.346 099	0.074 938
40	20	20	100	8	0.157 723	0	363	175.0 498	362	1296	373	0.361 891	0.086 189
40	20	20	100	10	0.223 253	0	191	96.11 557	191	1334	420	0.314 139	0.049 417
40	20	20	100	8	0.154 189	0	310	136.3 736	309	1342	364	0.333 995	0.067 679
40	20	20	140	6	0.243 349	0	388	207.8 742	386	1446	307	0.323 983	0.097 183
40	20	20	140	6	0.211 797	0	380	189.2 46	379	1420	310	0.326 695	0.089 733
40	20	20	140	6	0.172 125	0	434	230.3 457	433	1318	262	0.345 256	0.114 429
40	20	20	140	6	0.129 198	0	315	142.3 557	314	1366	317	0.315 974	0.071 285
40	20	20	140	6	0.220 342	0	378	176.3 33	378	1418	307	0.325 725	0.083 848
40	20	20	140	6	0.131 085	0	354	197.9 399	354	1286	273	0.327 757	0.103 471
40	20	30	60	10	0.692 663	2	46	13.68 71	21	1384	668	0.332 048	0.006 596
40	20	30	60	10	0.572 034	2	39	15.48 537	12	1342	699	0.345 985	0.007 535
40	20	30	60	14	0.595 696	2	50	19.01 042	31	1266	620	0.339 239	0.009 906
40	20	30	60	10	0.785 075	0	90	32.92 49	82	1276	605	0.349 975	0.016 773
40	20	30	60	10	0.651 746	4	56	16.70 635	1	1370	634	0.316 078	0.008 316
40	20	30	60	10	0.497 454	0	56	14.14 245	14	1358	581	0.304 659	0.007 241
40	20	30	100	10	0.434 89	0	267	110.0 678	265	1404	398	0.320 755	0.053 25
40	20	30	100	12	0.766 302	0	208	110.9 192	206	1314	383	0.309 511	0.058 287
40	20	30	100	16	1.059 998	0	313	152.8 339	312	1372	405	0.343 226	0.073 161
40	20	30	100	10	0.590 086	0	336	172.8 058	336	1330	381	0.350 269	0.084 419
40	20	30	100	8	0.435 137	0	279	137.1 712	279	1370	378	0.324 124	0.067 672
40	20	30	100	12	0.595 751	4	312	147.5 174	311	1274	358	0.343 606	0.075 767
40	20	30	140	10	0.431 149	0	401	186.3 729	399	1308	262	0.335 703	0.094 654
40	20	30	140	10	0.383 452	0	381	192.9 563	380	1290	267	0.334 022	0.099 616
40	20	30	140	14	0.659 669	0	426	200.7 243	426	1354	265	0.337 897	0.098 154
40	20	30	140	12	0.698 262	0	353	172.0 871	353	1438	272	0.302 957	0.083 416
40	20	30	140	8	0.398 442	0	335	158.8 862	335	1260	286	0.330 144	0.084 469
40	20	30	140	14	0.548 948	0	419	216.1 733	416	1310	259	0.340 05	0.108 903

40	20	40	60	26	2.964 046	0	31	9.799 616	2	1394	647	0.317 67	0.004 797
40	20	40	60	14	1.947 091	0	48	21.24 084	18	1364	613	0.316 291	0.010 647
40	20	40	60	16	1.664 016	0	106	48.32 362	74	1320	647	0.353 258	0.023 676
40	20	40	60	14	1.594 337	0	66	35.97 039	33	1292	644	0.343 829	0.018 268
40	20	40	60	20	1.378 584	0	56	27.99 327	22	1350	641	0.329 359	0.013 906
40	20	40	60	18	1.591 757	6	54	16.86 516	17	1364	576	0.302 089	0.008 592
40	20	40	100	18	1.987 672	0	268	138.4 487	268	1366	403	0.329 406	0.067 967
40	20	40	100	16	1.576 759	0	269	109.0 475	264	1352	409	0.332 346	0.053 851
40	20	40	100	20	1.684 93	0	288	138.7 026	285	1256	398	0.352 243	0.071 533
40	20	40	100	20	2.452 081	0	270	117.0 421	267	1398	402	0.323 657	0.056 624
40	20	40	100	16	1.738 028	2	180	87.39 466	180	1332	415	0.308 45	0.045 306
40	20	40	100	16	2.234 253	0	275	137.1 101	272	1440	391	0.315 264	0.065 197
40	20	40	140	16	1.579 126	0	352	185.1 515	352	1298	285	0.329 199	0.095 686
40	20	40	140	16	1.763 488	0	464	216.8 109	463	1344	270	0.352 913	0.104 387
40	20	40	140	18	2.186 364	0	353	184.6 499	353	1380	296	0.319 862	0.091 005
40	20	40	140	10	0.921 516	0	380	199.5 962	378	1324	269	0.328 26	0.101 266
40	20	40	140	16	1.595 857	0	389	214.5 04	388	1348	279	0.331 017	0.106 454
40	20	40	140	14	1.558 852	0	390	203.9 326	388	1298	291	0.343 45	0.103 153
40	30	20	60	6	0.096 548	0	166	88.80 601	166	1056	663	0.439 788	0.047 112
40	30	20	60	6	0.157 03	0	342	184.8 64	335	1130	618	0.457 513	0.088 749
40	30	20	60	6	0.095 997	0	157	86.82 928	148	1182	659	0.405 732	0.043 655
40	30	20	60	4	0.104 035	0	247	111.1 592	242	1090	647	0.449 217	0.056 169
40	30	20	60	8	0.131 311	0	133	49.77 914	130	1176	619	0.389 091	0.025 859
40	30	20	60	8	0.121 67	0	229	121.4 167	225	1126	658	0.439 522	0.060 436
40	30	20	100	6	0.086 967	0	504	257.3 765	499	1130	384	0.438 649	0.127 857
40	30	20	100	6	0.102 34	0	498	223.9 022	498	1188	393	0.428 571	0.107 697
40	30	20	100	6	0.117 398	0	469	266.9 953	469	1158	370	0.420 13	0.133 698
40	30	20	100	4	0.097 712	0	479	255.0 394	474	1114	417	0.444 389	0.127 202

40	30	20	100	6	0.126 357	0	418	214.2 334	418	1152	377	0.408 32	0.110 033
40	30	20	100	6	0.130 515	0	402	185.4 015	398	1180	423	0.410 295	0.092 654
40	30	20	140	8	0.115 572	0	584	283.3 744	584	1124	299	0.439 96	0.141 193
40	30	20	140	6	0.079 792	0	548	282.7 927	548	1108	293	0.431 503	0.145 096
40	30	20	140	6	0.083 919	0	557	282.1 568	556	1100	295	0.436 187	0.144 622
40	30	20	140	4	0.122 92	0	605	271.4 268	602	1146	275	0.433 515	0.134 17
40	30	20	140	4	0.067 441	0	523	246.2 458	523	1116	310	0.427 399	0.126 345
40	30	20	140	6	0.093 323	0	654	329.8 209	652	1126	267	0.449 389	0.161 282
40	30	30	60	10	0.522 889	0	242	142.8 166	238	1152	657	0.437 225	0.069 769
40	30	30	60	8	0.304 899	0	241	124.7 513	233	1134	660	0.440 553	0.061 545
40	30	30	60	12	0.319 447	2	390	195.5 084	390	1132	641	0.476 212	0.090 304
40	30	30	60	12	0.544 342	2	216	80.34 275	216	1152	635	0.424 439	0.040 071
40	30	30	60	8	0.468 173	2	208	87.20 194	198	1152	675	0.430 686	0.043 02
40	30	30	60	8	0.348 962	0	175	99.39 846	152	1102	687	0.432 251	0.051 21
40	30	30	100	8	0.345 598	0	490	233.9 098	488	1128	411	0.443 513	0.115 397
40	30	30	100	10	0.390 614	0	424	231.7 066	424	1164	411	0.417 709	0.115 911
40	30	30	100	10	0.306 763	0	495	247.5	494	1138	387	0.436 355	0.122 585
40	30	30	100	16	0.549 031	0	393	203.9 856	393	1172	454	0.419 515	0.101 033
40	30	30	100	8	0.407 235	0	527	266.3 33	527	1136	392	0.447 202	0.129 602
40	30	30	100	10	0.361 457	4	429	230.3 974	426	1130	389	0.418 163	0.118 213
40	30	30	140	8	0.424 304	0	537	266.8 702	537	1164	286	0.414 192	0.134 308
40	30	30	140	10	0.381 314	0	495	244.5 85	492	1154	313	0.410 924	0.124 852
40	30	30	140	10	0.353 969	0	597	296.5 188	592	1136	303	0.440 67	0.145 996
40	30	30	140	8	0.341 865	0	606	275.9 012	605	1070	292	0.456 024	0.140 265
40	30	30	140	10	0.378 067	0	510	259.9 775	510	1124	305	0.420 32	0.134 078
40	30	30	140	6	0.325 287	0	608	307.9 941	608	1090	311	0.457 442	0.153 307
40	30	40	60	12	0.909 508	8	255	144.1 998	253	1120	662	0.447 871	0.070 582
40	30	40	60	10	0.891 807	0	245	122.8 956	245	1120	646	0.443 063	0.061 112

40	30	40	60	10	0.923 928	0	210	93.65 841	201	1152	652	0.425 436	0.046 712
40	30	40	60	12	1.103 503	0	207	103.8 323	200	1172	621	0.411 942	0.052 099
40	30	40	60	10	1.009 457	4	165	67.82 595	163	1192	672	0.411 128	0.033 395
40	30	40	60	20	1.678 691	0	217	115.3 447	214	1178	663	0.426 764	0.056 129
40	30	40	100	10	0.940 481	0	466	236.9 464	465	1158	400	0.427 583	0.117 126
40	30	40	100	10	0.682 366	2	557	302.1 352	553	1118	390	0.457 101	0.146 454
40	30	40	100	12	1.196 283	0	466	236.9 779	466	1180	375	0.416 131	0.117 258
40	30	40	100	18	1.045 477	0	502	242.2 791	498	1166	381	0.429 829	0.118 474
40	30	40	100	18	1.149 636	0	454	246.0 9	452	1134	461	0.446 019	0.120 22
40	30	40	100	12	0.927 475	0	520	270.1 941	520	1162	399	0.441 615	0.129 839
40	30	40	140	20	1.499 43	0	505	270.3 206	503	1214	298	0.397 519	0.134 154
40	30	40	140	18	1.279 221	0	603	339.9 86	602	1080	265	0.445 3	0.174 62
40	30	40	140	10	0.886 394	2	620	303.6 282	617	1176	286	0.433 926	0.145 905
40	30	40	140	14	0.987 775	0	451	193.5 058	448	1140	311	0.399 684	0.101 899
40	30	40	140	14	1.195 677	2	598	310.3 893	598	1112	263	0.435 949	0.157 159
40	30	40	140	12	1.039 405	4	605	314.0 602	600	1150	253	0.425 012	0.156 482
40	40	20	60	4	0.060 958	0	281	126.5 021	273	986	678	0.490 965	0.065 308
40	40	20	60	4	0.053 897	0	311	166.3 586	311	936	616	0.497 585	0.089 296
40	40	20	60	4	0.080 721	0	309	160.0 876	308	1012	641	0.483 937	0.081 636
40	40	20	60	6	0.063 968	0	338	153.8 759	337	1000	658	0.498 747	0.077 131
40	40	20	60	6	0.107 952	0	359	172.6 163	358	1040	627	0.486 42	0.085 243
40	40	20	60	4	0.074 825	0	324	191.8 534	322	936	661	0.512 246	0.099 976
40	40	20	100	4	0.070 845	0	574	272.3 759	574	940	409	0.511 18	0.141 641
40	40	20	100	6	0.083 31	0	600	289.4 707	598	974	389	0.503 315	0.147 614
40	40	20	100	6	0.076 297	0	586	283.6 424	586	1044	379	0.480 338	0.141 186
40	40	20	100	4	0.046 011	0	546	278.3 759	546	970	405	0.495 055	0.144 912
40	40	20	100	6	0.076 184	0	571	267.6 119	567	1002	406	0.492 658	0.135 5
40	40	20	100	4	0.068 957	0	654	305.2 079	654	982	369	0.510 224	0.152 223

40	40	20	140	2	0.053 952	0	703	341.9 738	703	1014	294	0.495 773	0.170 052
40	40	20	140	6	0.097 495	0	759	376.7 1	758	1020	301	0.509 38	0.181 198
40	40	20	140	8	0.087 161	0	708	360.2 543	708	998	277	0.496 722	0.181 671
40	40	20	140	4	0.068 07	0	623	300.7 537	623	1070	286	0.459 323	0.151 973
40	40	20	140	6	0.081 338	0	676	367.0 617	676	1056	273	0.473 317	0.183 073
40	40	20	140	6	0.085 343	4	741	385.8 876	741	1056	286	0.492 094	0.184 901
40	40	30	60	8	0.239 97	0	427	209.5 086	419	1004	628	0.510 483	0.102 149
40	40	30	60	6	0.218 664	2	331	198.4 795	327	1008	676	0.498 261	0.098 599
40	40	30	60	8	0.278 818	0	382	170.3 079	376	1002	635	0.502 235	0.084 604
40	40	30	60	6	0.209 529	0	352	167.2 802	346	992	653	0.501 758	0.084 018
40	40	30	60	6	0.242 971	0	287	160.2 412	285	988	632	0.481 365	0.084 116
40	40	30	60	6	0.206 396	0	277	152.7 112	274	1040	689	0.480 779	0.076 241
40	40	30	100	6	0.184 021	0	668	316.1 952	667	984	410	0.522 562	0.153 418
40	40	30	100	8	0.372 282	0	543	274.4 86	543	986	414	0.492 537	0.141 269
40	40	30	100	6	0.244 089	0	672	336.8 45	670	1032	371	0.502 171	0.162 492
40	40	30	100	6	0.213 461	0	649	313.2 111	647	984	402	0.515 986	0.154 063
40	40	30	100	8	0.241 78	0	656	331.9 761	656	974	383	0.516 145	0.164 916
40	40	30	100	6	0.209 871	0	462	202.9 63	462	1000	393	0.460 916	0.109 414
40	40	30	140	14	0.223 98	0	663	325.7 366	661	956	326	0.507 977	0.167 646
40	40	30	140	8	0.256 52	2	655	313.1 844	653	988	292	0.488 372	0.161 852
40	40	30	140	14	0.419 831	0	648	327.3 83	648	1000	271	0.478 895	0.170 601
40	40	30	140	8	0.347 663	0	815	435.8 329	815	1020	264	0.514 054	0.207 638
40	40	30	140	10	0.327 951	0	704	343.9 951	702	1060	281	0.481 155	0.168 377
40	40	30	140	12	0.276 386	0	741	407.5 584	739	984	294	0.512 147	0.202 062
40	40	40	60	10	0.518 684	0	252	123.4 122	249	958	674	0.490 696	0.065 61
40	40	40	60	12	0.506 512	0	318	140.1 351	317	1050	658	0.481 481	0.069 202
40	40	40	60	14	0.599 439	0	285	170.2 354	285	1054	644	0.468 482	0.085 847
40	40	40	60	10	0.539 23	0	358	162.5 612	357	984	640	0.503 281	0.082 06

40	40	40	60	8	0.617 08	0	364	178.3 164	362	1066	665	0.490 683	0.085 197
40	40	40	60	8	0.404 109	0	365	158.7 133	360	930	671	0.525 752	0.080 935
40	40	40	100	8	0.520 829	0	584	289.7 719	583	996	400	0.496 716	0.146 423
40	40	40	100	16	0.695 829	0	657	331.1 816	657	992	368	0.508 18	0.164 195
40	40	40	100	10	0.615 964	0	675	314.1 715	673	1046	362	0.497 357	0.150 971
40	40	40	100	10	0.519 929	0	604	277.6 313	604	956	395	0.510 997	0.142 011
40	40	40	100	22	0.795 697	14	665	346.0 285	665	1010	418	0.514 001	0.164 228
40	40	40	100	12	0.559 862	0	617	323.0 741	616	960	393	0.512 443	0.164 08
40	40	40	140	8	0.485 326	0	744	379.7 438	744	948	265	0.515 585	0.194 044
40	40	40	140	14	0.697 805	0	768	418.0 604	762	968	291	0.521 029	0.206 858
40	40	40	140	12	0.814 112	0	814	397.7 072	814	990	301	0.529 691	0.188 935
40	40	40	140	10	0.643 322	0	759	368.6 422	759	1014	264	0.502 209	0.180 973
40	40	40	140	10	0.740 056	0	677	317.3 752	677	1066	286	0.474 618	0.156 42
40	40	40	140	12	0.576 471	0	750	370.7 315	750	988	287	0.512 099	0.183 077

Creat e Delay	FirstR eg Delay	Secon dReg Delay	FullRe g Delay	MaxQ 1	Mean Q1	LeftQ 1	MaxQ 2	Mean Q2	LeftQ 2	Num Of Secon d	Num OfFull	Fprob	Sload
30	30	30	100	10	0.533 473	0	935	496.7 474	932	1342	411	0.500 186	0.185 008
30	30	30	100	16	0.678 782	2	999	463.5 025	999	1306	384	0.513 935	0.172 242
30	30	30	100	10	0.463 884	0	882	462.1 824	882	1322	445	0.500 944	0.174 474
30	30	30	100	14	0.892 977	0	856	457.3 964	854	1382	343	0.464 133	0.177 354
30	30	30	100	16	0.662 255	0	864	414.0 339	863	1346	426	0.489 184	0.157 129
30	30	30	100	12	0.747 057	0	883	450.9 087	883	1328	392	0.489 819	0.173 227
30	30	30	100	12	0.481 134	0	904	469.2 357	903	1284	374	0.498 633	0.183 224
30	30	30	100	10	0.643 223	0	947	461.8 62	946	1344	407	0.501 669	0.171 25
30	30	30	100	12	0.585 303	0	1005	501.9 321	1005	1326	404	0.515 174	0.183 522
30	30	30	100	8	0.450 623	0	885	449.4 071	885	1260	396	0.504 132	0.176 862

30	30	30	100	8	0.526 168	0	999	497.8 306	999	1322	368	0.508 367	0.185 136
30	30	30	100	12	0.623 9	0	930	455.6 263	930	1338	423	0.502 787	0.169 315
30	30	30	100	10	0.695 685	0	1040	498.2 367	1040	1350	431	0.521 446	0.176 617
30	30	30	100	24	0.845 356	0	838	424.0 96	838	1332	405	0.482 718	0.164 697
30	30	30	100	10	0.572 801	0	864	437.1 871	860	1292	419	0.497 472	0.170 046
30	30	30	100	12	0.503 196	0	912	466.5 077	912	1276	375	0.502 146	0.182 016
30	30	30	100	10	0.477 743	2	856	414.3 8	856	1310	423	0.493 632	0.159 931
30	30	30	100	16	0.557 061	0	904	422.0 929	903	1338	396	0.492 605	0.160 066
30	30	30	100	10	0.581 571	0	949	465.6 436	947	1300	394	0.507 762	0.176 313
30	30	30	100	16	0.640 48	0	982	503.2 878	982	1250	415	0.527 767	0.190 135
30	30	30	100	10	0.604 825	0	996	484.9 455	996	1304	413	0.519 351	0.178 749
30	30	30	100	12	0.733 656	0	1012	486.4 837	1012	1334	383	0.511 176	0.178 264
30	30	30	100	14	0.889 12	0	877	447.1 534	877	1320	446	0.500 568	0.169 184
30	30	30	100	12	0.576 401	0	862	465.5 234	862	1270	409	0.500 197	0.183 205
30	30	30	100	10	0.554 661	0	900	476.1 791	899	1330	390	0.492 173	0.181 817
30	30	30	100	20	0.770 095	2	953	439.2 545	949	1362	446	0.505 618	0.159 208
30	30	30	100	8	0.498 16	0	982	476.0 689	981	1304	408	0.515 782	0.176 78
30	30	30	100	12	0.681 91	0	961	454.6 676	960	1350	387	0.499 444	0.168 583
30	30	30	100	12	0.463 87	0	898	421.7 222	897	1318	388	0.493 661	0.162 014
30	30	30	100	12	0.530 804	0	1016	520.2 796	1016	1324	423	0.520 811	0.188 302
30	30	30	100	8	0.539 339	0	898	429.9 877	898	1372	383	0.482 85	0.162 076
30	30	30	100	8	0.511 164	0	881	402.6 104	881	1302	394	0.494 761	0.156 232
30	30	30	100	14	0.835 197	0	989	486.6 239	988	1346	389	0.505 692	0.178 709
30	30	30	100	8	0.551 105	0	952	474.7 939	952	1342	407	0.503 147	0.175 784
30	30	30	100	14	0.543 82	2	982	519.8 403	981	1264	374	0.516 978	0.198 337
30	30	30	100	10	0.617 124	0	906	437.0 513	902	1298	377	0.496 314	0.169 597
30	30	30	100	12	0.559 104	0	989	505.1 335	989	1374	394	0.501 632	0.183 219
30	30	30	100	12	0.405 115	0	897	453.8 194	897	1290	406	0.502 507	0.175 017

30	30	30	100	10	0.590 494	0	903	439.1 188	903	1372	394	0.485 95	0.164 526
30	30	30	100	12	0.835 107	0	862	453.7 757	862	1294	393	0.492 35	0.178 021
30	30	30	100	12	0.556 776	0	952	480.4 291	952	1338	401	0.502 787	0.178 532
30	30	30	100	16	0.726 363	0	849	437.9 492	846	1306	365	0.481 128	0.173 997
30	30	30	100	10	0.437 454	0	888	459.5 908	888	1264	401	0.504 896	0.180 02
30	30	30	100	10	0.641 211	0	912	472.9 656	912	1346	409	0.495 313	0.177 34
30	30	30	100	10	0.567 131	0	1034	477.7 59	1031	1312	392	0.520 293	0.174 683
30	30	30	100	12	0.571 45	2	973	453.7 021	973	1318	380	0.506 173	0.169 735
30	30	30	100	14	0.890 062	4	1001	468.1 992	1001	1304	404	0.517 877	0.172 576
30	30	30	100	10	0.665 589	0	841	420.3 484	840	1354	373	0.472 536	0.163 751
30	30	30	100	8	0.494 358	0	788	387.9 274	788	1304	415	0.479 856	0.154 738
30	30	30	100	14	0.668 09	0	826	378.2 96	821	1330	386	0.475 759	0.149 112
30	30	30	100	10	0.455 92	0	922	471.3 096	922	1268	393	0.509 098	0.182 466
30	30	30	100	10	0.570 989	0	941	461.4 729	938	1342	383	0.496 057	0.173 291
30	30	30	100	10	0.502 797	2	946	464.7 573	946	1324	399	0.503 557	0.174 001
30	30	30	100	8	0.454 313	0	929	446.3 363	928	1324	371	0.495 234	0.170 163
30	30	30	100	10	0.607 09	6	936	462.3 234	935	1346	402	0.497 211	0.171 931
30	30	30	100	12	0.856 824	0	853	442.5 559	853	1376	402	0.477 005	0.168 208
30	30	30	100	10	0.552 754	2	977	492.8 779	977	1354	374	0.499 076	0.182 075
30	30	30	100	10	0.588 637	6	959	425.6 67	956	1288	409	0.513 351	0.160 085
30	30	30	100	10	0.522 669	0	986	477.8 268	986	1306	375	0.510 311	0.179 163
30	30	30	100	12	0.706 868	0	892	454.9 413	892	1330	389	0.490 617	0.174 24
30	30	30	100	12	0.650 03	0	948	493.7 8	948	1298	407	0.510 743	0.186 121
30	30	30	100	8	0.469 491	0	919	453.8 642	919	1386	406	0.488 75	0.167 416
30	30	30	100	14	0.680 558	0	916	473.4 643	916	1382	397	0.487 199	0.175 682
30	30	30	100	16	0.618 92	0	925	452.6 236	925	1290	428	0.511 918	0.171 254
30	30	30	100	12	0.764 697	0	851	400.4 134	851	1350	460	0.492 672	0.150 475
30	30	30	100	12	0.597 282	0	991	480.5 214	991	1360	402	0.505 993	0.174 545

30	30	30	100	10	0.565 95	0	856	446.6 961	853	1320	420	0.490 937	0.172 27
30	30	30	100	10	0.488 205	2	846	443.1 636	846	1296	413	0.492 374	0.173 314
30	30	30	100	10	0.599 954	0	1074	523.5 394	1074	1346	393	0.521 507	0.186 114
30	30	30	100	10	0.554 243	2	974	497.8 309	973	1322	378	0.505 047	0.186 105
30	30	30	100	10	0.531 254	0	955	455.9 64	951	1360	376	0.493 859	0.169 693
30	30	30	100	10	0.503 671	0	1003	464.6 733	1003	1294	382	0.516 984	0.173 45
30	30	30	100	14	0.707 413	0	921	465.9 654	921	1364	404	0.492 748	0.173 286
30	30	30	100	10	0.447 671	0	922	438.2 543	922	1352	375	0.489 619	0.165 441
30	30	30	100	8	0.593 107	0	901	449.5 755	900	1312	381	0.494 022	0.173 38
30	30	30	100	12	0.642 657	0	1092	528.7 845	1092	1386	385	0.515 892	0.184 696
30	30	30	100	10	0.518 739	0	945	460.7 577	945	1362	380	0.493 115	0.171 477
30	30	30	100	12	0.551 302	0	971	467.5 121	971	1290	376	0.510 808	0.177 289
30	30	30	100	10	0.444 554	0	1027	498.2 86	1025	1290	406	0.525 91	0.183 126
30	30	30	100	12	0.647 635	0	908	469.2 993	908	1332	405	0.496 408	0.177 429
30	30	30	100	12	0.591 757	0	796	392.2 407	794	1348	415	0.472 82	0.153 399