## Опис математичної моделі вибору оптимального складу компютеру

Існує *n* типів деталей та mi деталей кожного з n типів. Для кожної деталей відомі наступні параметри:

*s ij* – ціна *j*-ої деталі *i-ого* типу;

*q ij* –якість *j*-ої деталі *i-ого* типу;

*p ij* –потужність *j*-ої деталі *i-ого* типу;

*Li* – множина деталей для замовлення;

| *Li* | *= m –* кількість деталей, що необхідно зібрати;

Для кожного замовлення відомо:

S – максимальну ціну замовлення.

Q – середню якість деталей замовлення.

P – середню потужність деталі.

Тоді задача зводиться до знаходження min та маємо наступні обмеження:

## Алгоритм рішення

### Опис алгоритму

Зважаючи на те, що для рішення цієї задачі методом повного перебору необхідно занадто багато часу, ( наприклад, для 5 видів деталей, по 120 деталей кожного типу = 24883200000 ) було вирішено реалізувати для цієї еврестичний алгоритм. А саме генетичний алгоритм.

Для опису генетичного алгоритму введемо наступні поняття:

Хромосома – певна деталь яка входить до складу кaопьютера.

Особина – набір деталей (хромосом) певного типу характерного для данного замовлення.

Популяція – набір компютерів (осіб) різного складу, які задовільняють початковим умовам.

Якщо особина задовольняє умовам поставленої задачі то будемо рахувати, що особина жива.

Можна виділити такі етапи генетичного алгоритму:

1. Створення початкової популяції:
2. Повторювання до виконання критерію зупинки алгоритму:
   1. Вибір індивідів із поточної популяції (селекція)
   2. Схрещення або/та мутація
   3. Обчислення функції пристосовуваності для всіх осіб
   4. Формування нового покоління

Для початку роботи алгоритму необхідно згенерувати початковий набір особин – початкову популяцію. Для цього згенеруємо 20 випадкових особин. Якщо за 500 спроб невдалося згенерувати жодної живої особини, то будемо рахувати, що задача не має ріщення.

Після створення початкової популяції будемо виконувати 1000 кроків еволюції для покращення популяції.

На кожному кроці єволюції обираємо дві особини з поточної популяції. Випадковим чином вибираємо точку схрещування та утворюємо масив з двох особин нащадків. Для кожного з нащадків перевіряємо чи живі назадки. Якщо так, то заносимо нащадка до популяції та видаляємо з популяції найгіршу особину. Якщо нащадок не задовольняє умовам популяції обираємо з його деталей (хромосом) таку *i-ту*, для якої:

Цю хромосому ми змінюємо на випадкову іншу деталь з набору відповідних хромосом. Якщо після реанімації особина задовольняє умовам, то повертаємо її у популяцію.

Для порівняя двох особин з однаковими значеннями цільової функції обираємо ту у якої сума критеріїв якості та потужності більша.

Умовою припинення еволюційних кроків є вичерпання числа поколінь, що відпущені на еволюцію. У нашому випадку 1000 кроків.

Після отримання кінцевої популяції обераємо з неї особину яка найбільше відповідає критеріям виживання (має найменший показник для цільової функції )

### Тестування алгоритму.

Для тестування алгоритму була створена база данних деталей, налічувала 5 типів деталей та 10 деталей кожного типу та проведено 3 типи випробувань:

1. Задана ціна у 2000 одиниць та показники якості та потужності 2 та 1 відповідно.
2. Задана ціна у 4000 одиниць та показники якості та потужності 3 та 3 відповідно.
3. Задана ціна у 5000 одиниць на показники якості та потужності 4 та 4 відповідно.

Для кожного варіанту тестування тест проводився 10 разів.

Результати першого тесту наведені у таблиці 1

Таблиця 1

|  |  |  |  |
| --- | --- | --- | --- |
| № | Якість | Потужність | Ціна |
| 1 | 3.0 | 2.6 | 840 |
| 2 | 3.6 | 1.4 | 1100 |
| 3 | 3.0 | 2.6 | 840 |
| 4 | 2.2 | 3.4 | 840 |
| 5 | 3.6 | 2.2 | 1250 |
| 6 | 3.2 | 2.8 | 940 |
| 7 | 2.4 | 3.6 | 940 |
| 8 | 2.2 | 3.4 | 840 |
| 9 | 2.2 | 3.4 | 840 |
| 10 | 2.6 | 2.0 | 940 |

Методом повного перебору був знайдений найкращій результат: 840. Для цього випадку алгоритм в середньому давав відповідь на 11% менш вигідну ніж оптимальна ціна.

Для двох інших випадків було отримано результати на 33% та 10% відповідно. Результати зображені на рисунку 1.

Рисунок 1 – відношення абсолютних розвязків до розязків знайдених генетичним алгоритмом

У середньому наша реалізація генетичного алгоритму на тестових данних давала не більше 20% відхилення від результату знайденим методом повного перебору, при цьому зменшуючи кількість ітерацій порівняння з 105 до 103, тобто у 100 разів.