

# Разработка алгоритмов оптимизации групповой работы мобильных роботов

Студент группы с8503а, Спорышев М. С.

Руководитель:

н.с. лаборатории необитаемых подводных аппаратов и их систем, к.т.н.  
Туфанов И. Е.

25 Июня 2015

- АНПА – автономный необитаемый подводный аппарат
- СПУ – система программного управления
- ГА – генетический алгоритм

## Обзорно поисковые задачи

- Поиск затонувших объектов
- Замеры параметров водной среды
- Исследование локальных неоднородностей

## Использование групп АНПА

- Централизованное управление
- Групповое управление

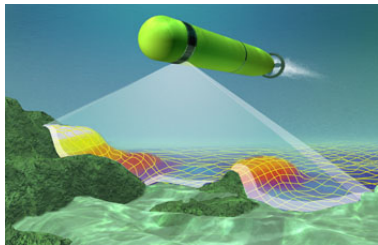
- 1 Составление заданий для аппаратов
- 2 Распределение заданий между аппаратами
- 3 Перепланирование

# Составление заданий для аппаратов

Составляются  $n$  заданий. Предполагается, что каждый аппарат может выполнять каждое задание.

$i$ -е задание может быть выполнено в одном из  $v_i$  вариантов:

- $(a_{i1}, b_{i1}, \tau_{i1})$
- $(a_{i2}, b_{i2}, \tau_{i2})$
- ...
- $(a_{iv_i}, b_{iv_i}, \tau_{iv_i})$



# Распределение заданий между аппаратами

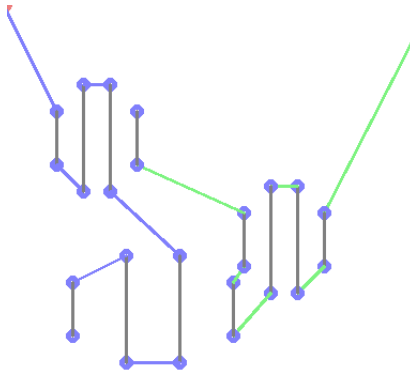
Имеется  $m$  аппаратов. Каждому аппарату составляется план заданий так, чтобы каждое было выполнено в единственном варианте, только одним аппаратом, единственный раз.

Результат распределения заданий:

- $q$ -му аппарату соответствует последовательность пар номеров заданий и их вариантов:

$$p_q = ((i_1, j_1), (i_2, j_2), \dots, (i_{|p|}, j_{|p|}))$$

- $\forall k \in 1..|p|$ 
  - $i_k \in 1..n$  – номер задания
  - $j_k \in 1..v_{i_k}$  – номер варианта



- Выход аппарата из строя
- Появление нового аппарата
- Появление новых заданий
- Изменение заданий

## Алгоритм Хельда-Карпа

- Экспоненциальная зависимость времени работы от количества заданий (Перебор подмножеств)
- Работает дольше минуты для 20 заданий

### Цель

Разработать новые алгоритмы для решения задачи планирования, способные составлять планы для 100 заданий и 5 аппаратов за приемлемое время(до минуты) и внедрить их в существующую СПУ.



## Входные данные

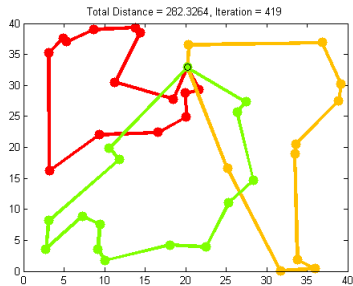
- Имеется  $m$  аппаратов и  $n$  заданий.  $q$ -ый аппарат в начальный момент времени находится в точке  $s_q$
- $d_q(\mathbf{a}, \mathbf{b}) = |\mathbf{a} - \mathbf{b}|/u_q$  – время перехода АНПА от точки  $\mathbf{a}$  к точке  $\mathbf{b}$ , где  $u_q$  - максимальная скорость  $q$ -го аппарата.
- План  $q$ -го аппарата:  $p = ((i_1, j_1), (i_2, j_2), \dots, (i_{|p|}, j_{|p|}))$ .
- $t_q(p) = d_q(\mathbf{s}_q, \mathbf{a}_{i_1, j_1}) + \sum_{k=2}^{|p|} d_q(\mathbf{b}_{i_{k-1} j_{k-1}}, \mathbf{a}_{i_k j_k}) + \sum_{k=1}^{|p|} \tau_{i_k j_k}$
- Общий план  $P = (p_1, p_2, \dots, p_m)$
- Время выполнения общего плана:  $t(P) = \max_{q \in 1..m} t_q(p)$

## Задача

Найти общий план  $P$  при котором  $t(P) \rightarrow \min$

## Множественная задача коммивояжера

- Каждое задание является единственной точкой
- Полный граф
- Стартовая вершина
- Поиск оптимальной системы циклов
- Минимизация суммы, максимума



```
1: for all  $t \in T$  do
2:   for all  $var \in Vars(t)$  do
3:     for all  $v \in V$  do
4:        $time, pos \leftarrow MINPATHTIME(t, var, Plan(v))$ 
5:       if  $time < minTime$  then
6:          $minTime \leftarrow time$ 
7:          $bestPos \leftarrow pos$ 
8:          $bestV \leftarrow v$ 
9:          $bestVar \leftarrow var$ 
10:      end if
11:    end for
12:  end for
13:   $INSERT(t, bestVar, Plan(bestV), bestPos)$ 
14:   $OPTIMIZEVARS(bestV)$ 
15: end for
```

# Оптимальный выбор варинатов

Известен план некоторого аппарата, найти варианты заданий, дающие минимальное время выполнения этого плана.

Метод динамического программирования

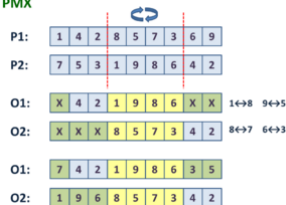
- Номер последнего задания
- Номер варианта задания

Переход к следующему состоянию

```
 $val \leftarrow d[i-1][var1] + Dist(v, p[i-1], var1, p[i], var2)$   
if  $d[i][var2] > val$  then  
     $d[i][var2] \leftarrow Min(d[i][var2], val)$   
     $prev[i][var2] = var1$   
end if
```

- Решение представлено в виде двух хромосом
  - Перестановка номеров заданий
  - Последовательность номеров аппаратов
- 3 вида мутаций
  - Swap
  - Reverse
  - Select
- Скращивание – Partially matched crossover
- Функция приспособленности

PMX



- Точное решение
- TSP – сотни заданий за секунды
- Множественная задача коммивояжера
  - Оптимизация максимума
  - Бинарный поиск максимума и оптимизация суммы с доп. ограничениями
- Работает медленнее алгоритма Хельда-Карпа

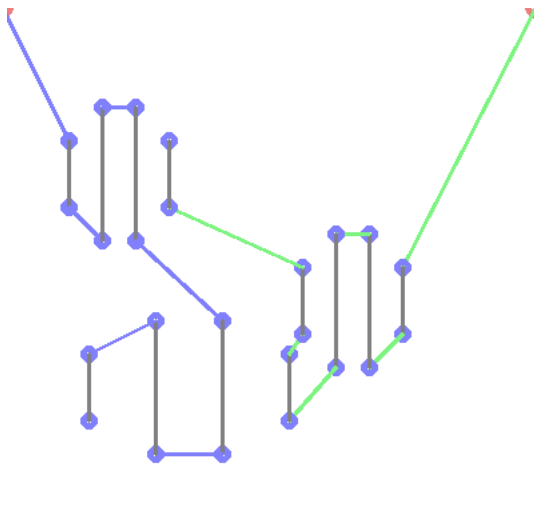
## Сторонние библиотеки

- FlopC++
- SYMPHONY
- OpenCV
- Boost

## Python

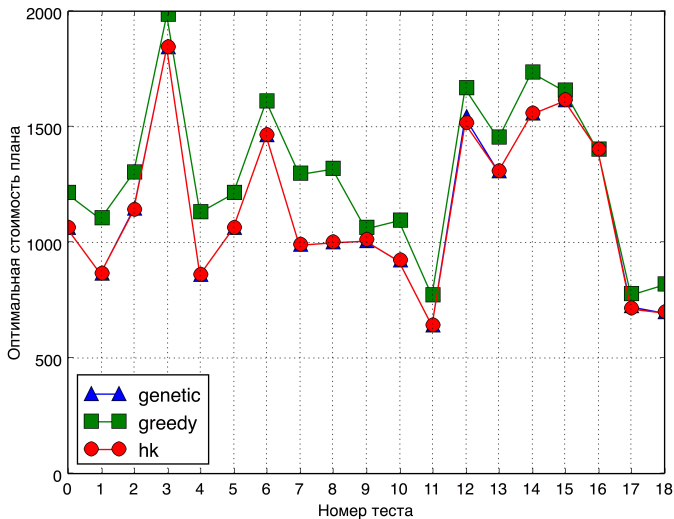
- Pandas
- matplotlib

## Визуализация

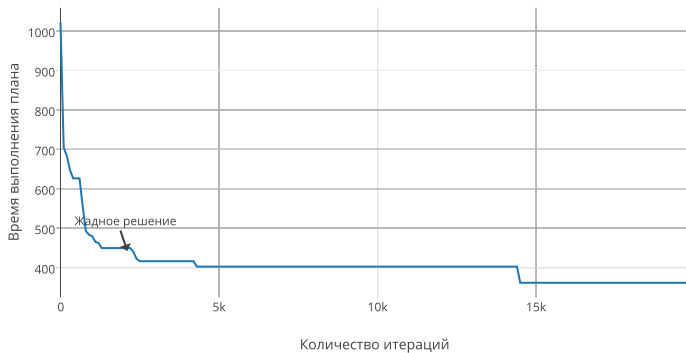




## Время выполнения плана на каждом тесте



## Стоимость решения ГА от количества итераций



Алгоритмы успешно внедрены

- до 18 заданий – алгоритм Хельда-Карпа
- до 40 заданий – генетический алгоритм
- от 40 заданий – жадный алгоритм

- Система контроля версий git, 52 коммита, +4900 -1800.  
Языки C++ и Python.