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.

Соруководитель:
старший преподаватель каф. информатики, математического и компьютерного моделирования
Кленин А. С.

25 Июня 2015

Термины

- АНПА – автономный необитаемый подводный аппарат
- СПУ – система программного управления
- ГА – генетический алгоритм
- МАРК – морской автономный робототехнический комплекс лаборатории НПА и их систем ДВФУ.

Автономные необитаемые подводные аппараты

Обзорно-поисковые задачи

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

Использование групп АНПА

- Централизованное управление
- Групповое управление

Используемая система управления

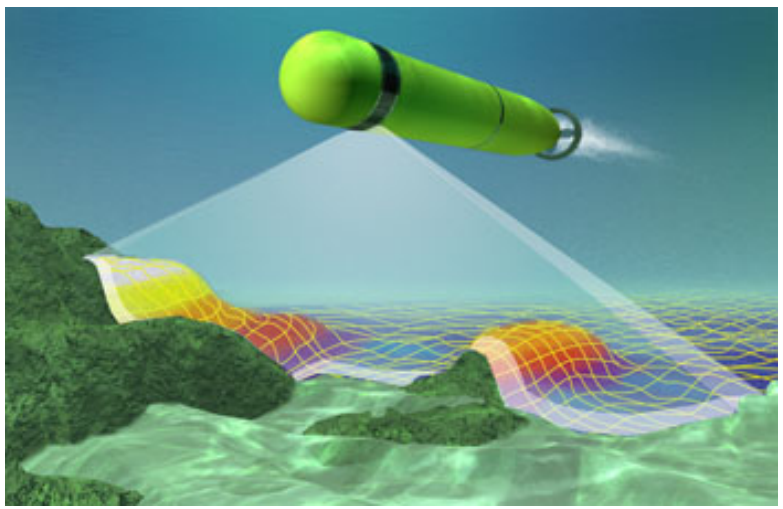
1. Составление заданий для аппаратов
2. Распределение заданий между аппаратами
3. Перепланирование

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:

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

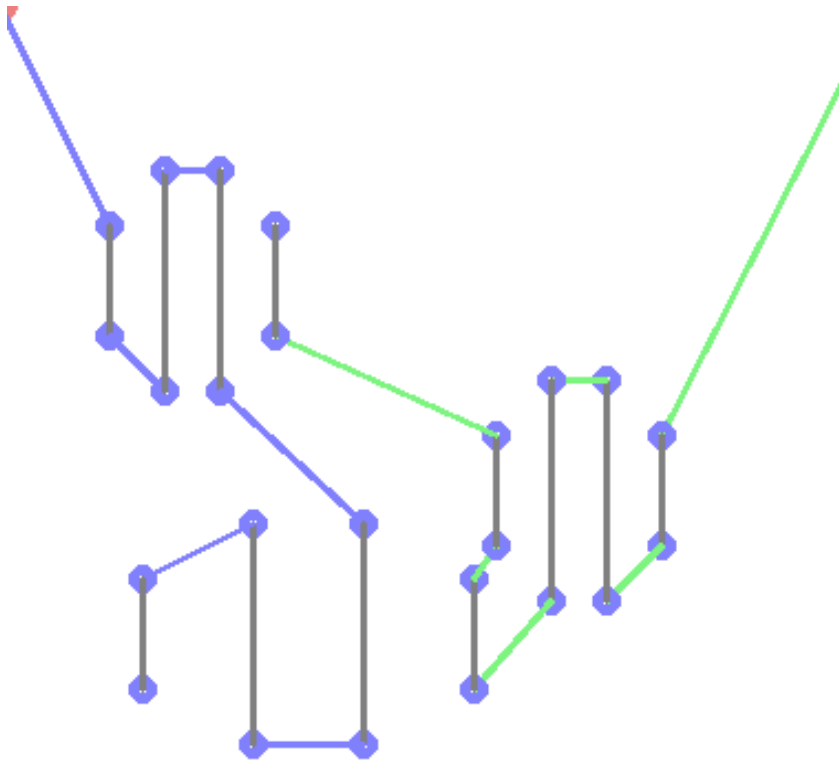


Распределение заданий между аппаратами

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: $p_q = ((i_1, j_1), (i_2, j_2), ..., (i_{|p_q|}, j_{|p_q|}))$
- $\forall k \in 1..|p_q|$
  - $i_k \in 1..n$ – номер задания
  - $j_k \in 1..v_{i_k}$ – номер варианта



Перепланирование

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

Цель работы

Раньше использовался 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(несколько минут) и внедрить их в существующую СПУ.

Математическая модель

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: $p = ((i_1, j_1), (i_2, j_2), ..., (i_{|p|}, j_{|p|}))$.
- $t_q(p) = d_q(\mathbf{s}_q, \mathbf{a}_{i_1,j_1}) + \sum\limits_{k=2}^{|p|} d_q(\mathbf{b}_{i_{k-1}j_{k-1}}, \mathbf{a}_{i_k j_k}) + \sum\limits_{k=1}^{|p|} \tau_{i_k j_k}$
- Общий план $P = (p_1, p_2, ..., p_m)$
- Время выполнения общего плана: $t(P) = \max\limits_{q \in 1..m} t_q(p)$

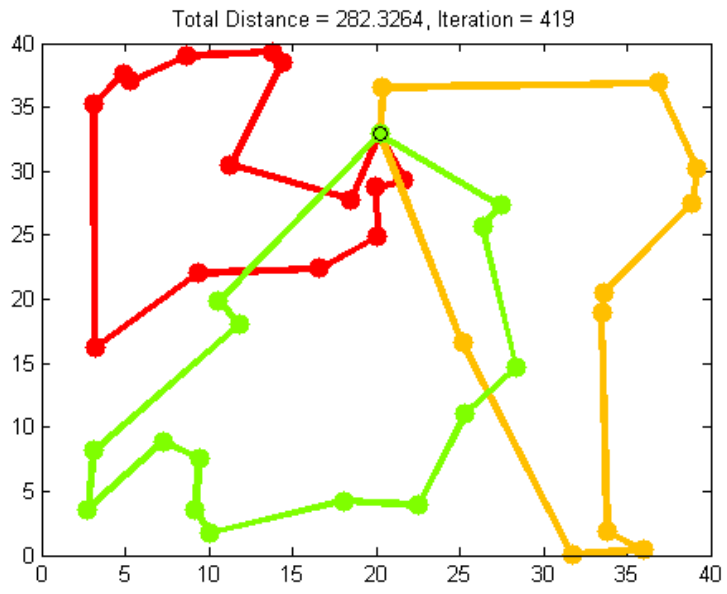
Задача

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

Существующие решения

Множественная задача коммивояжера

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



Жадный алгоритм

T – номера заданий. V – номера аппаратов. $Vars(t)$ - номера вариантов задания t.

1: **for all** $t \in T$ **do**
2: **for all** $var \in Vars(t)$ **do**
3: **for all** $v \in V$ **do**
4: $time, pos \leftarrow \textsc{MinPathTime}(t, var, Plan(v))$
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: $\textsc{Insert}(t, bestVar, Plan(bestV), bestPos)$
14: $\textsc{OptimizeVars}(bestV)$
15: **end for**

Оптимальный выбор вариантов

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

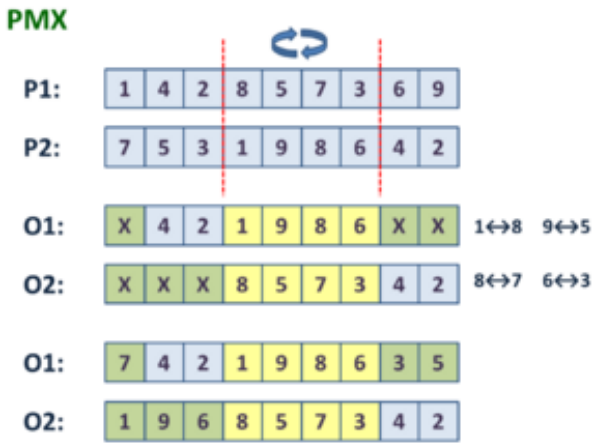
- Номер последнего задания
- Номер варианта задания

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

Генетический алгоритм

- Решение представлено в виде двух хромосом
  - Перестановка номеров заданий
  - Последовательность номеров аппаратов
- Используется 3 вида мутаций
  - Swap
  - Reverse
  - Select
- Оператор скрещивания – Partially matched crossover
- Функция приспособленности использует метод динамического программирования для оптимального подбора вариантов.



Целочисленное программирование

Плюсы подхода

- Находит точное решение в отличие от предыдущих методов
- Позволяет решать задачу коммивояжера для 100 заданий за секунды

Реализация

- Строим решение для упрощенной задачи (mTSP).
- Метод ветвей и границ для решения задачи дискретной оптимизации.
- Оптимизация максимума
- Бинарный поиск максимума и оптимизация суммы с доп. ограничениями

Результат

Работает медленнее алгоритма Хельда-Карпа

Реализация

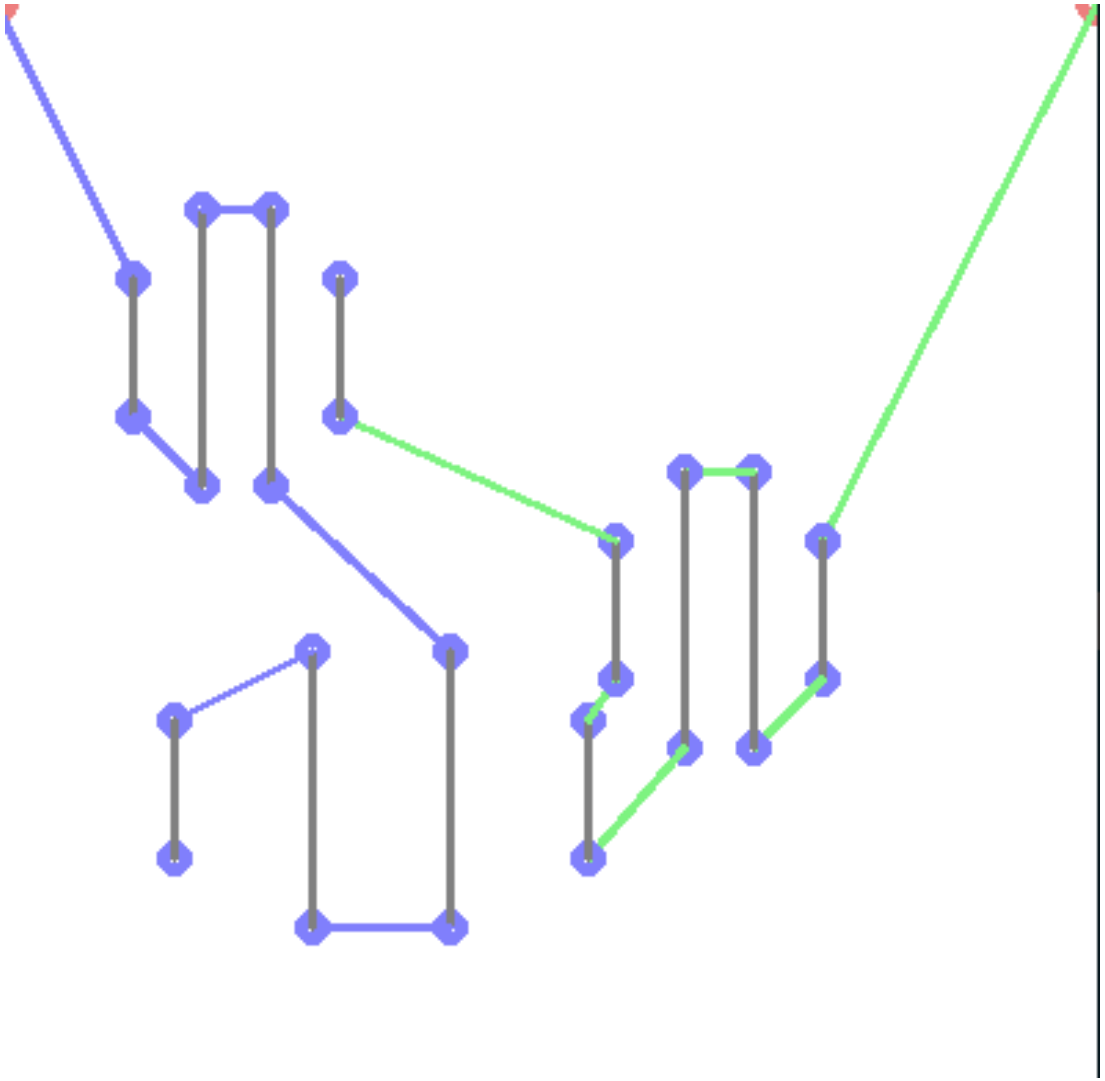
Стороннине библиотеки

- FlopC++ – формулировка задачи целочисленной оптимизации на C++
- SYMPHONY – фреймверк для решения задач целочисленного программирования
- OpenCV – визуализация полученных решений
- Boost – расширение стандартных возможностей C++

Python

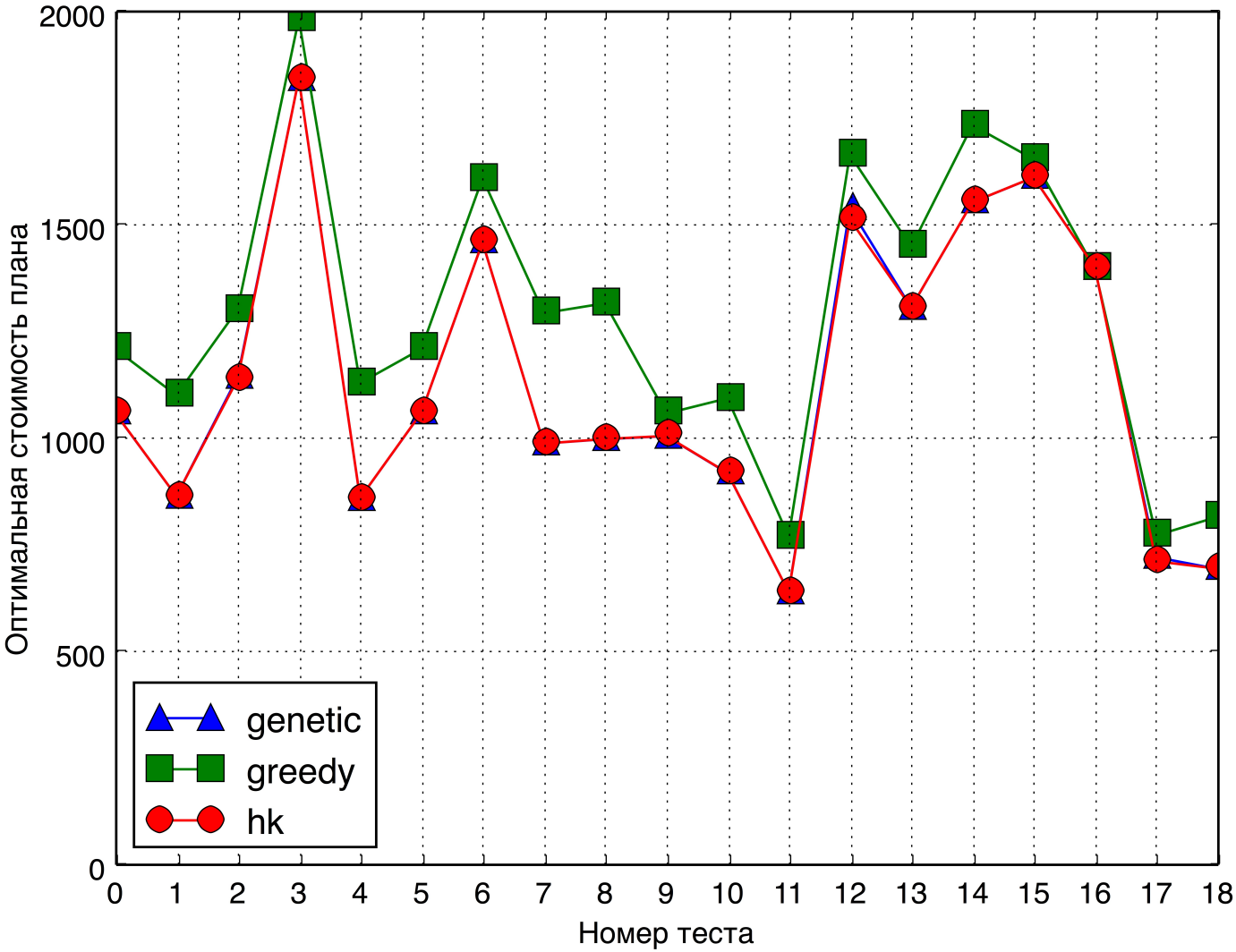
- Pandas – сохранение и загрузка данных
- matplotlib – построение графиков

Визуализация



Тестирование

Время выполнения плана на каждом тесте



Стоимость решения ГА от количества итераций

Результат

Алгоритмы внедрены в СПУ комплекса “МАРК”.

Автоматический выбор алгоритма в зависимости от размера задачи.

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

Заключение

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