Разработка алгоритмов оптимизации групповой работы мобильных роботов

Студент группы c8503a, Спорышев М. С.

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

25 Июня 2015

Термины

- АНПА – автономный необитаемый подводный аппарат
- СПУ – система программного управления

Зачем нужны АНПА

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 меджу аппаратами
   - Выбор последовательности заданий
   - Выбор варианта заданий
     - Точка начала
     - Точка конца
     - Время выполнения
3. Перепланирование
   - Выход аппарат из строя
   - Появление нового аппарата
   - Появление новых заданий
   - Изменение заданий

Математическая модель

- 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.
- Планом аппарата названа последовательность пар

$$p = ((i_1, j_1), (i_2, j_2), ..., (i_{|p|}, j_{|p|})), \forall k \in 1..|p|\ i_k \in 1..n, j_k \in 1..v_{i_k}$$

.
- Время выполнения аппаратом с номером q плана p, составляет:

$$t_q(p) = d_q(\mathbf{s}_q, \mathbf{a}_{i_1,j_1}) + \sum_{k=2}^{|p|} d_q(\mathbf{b}_{i_{k-1},j_{k-1}}, \mathbf{a}_{i_k,j_k}) + \sum_{k=1}^{|p|} l_{i_k,j_k}$$

- Общий план $P = (p_1, p_2, ..., p_m)$
- Время выполнения общего плана: $t(P) = \max\limits_{q \in 1..m} t_q(p_q)$
- $t(P) \to \min$

Существующие решения

Множественная задача коммивояжера

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

Жадный алгоритм

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 варинатов

Известна последовательность заданий, найти варианты заданий, дающие минимальное время выполнения такой последовательности.
Метод динамического программирования

- Номер последнего задания
- Номер варианта задания

$val \leftarrow d[i-1][var1] + Dist(p[i-1], p[i])$
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
- 3 вида мутаций
  - Swap
  - Reverse
  - Select
- Скрещивание – Partially matched crossover
- Функция приспособленности

Целочисленное программирование

- 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

Реализация

Стороннине библиотеки

- FlopC++
- SYMPHONY
- OpenCV
- Boost

Python

- Pandas
- matplotlib

Тестирование

Результат

Алгоритмы успешно внедрены

- 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.