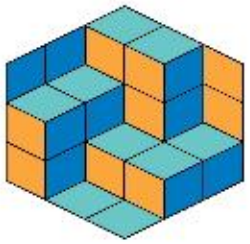
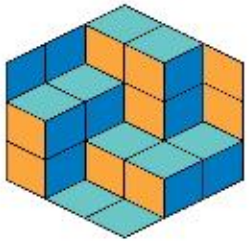


Алгоритмы конфликтно-ориентированного поиска для задачи много-агентного планирования.

CBS Basics and enhancements

Проект подготовлен в рамках курса
“Методы и алгоритмы эвристического поиска”

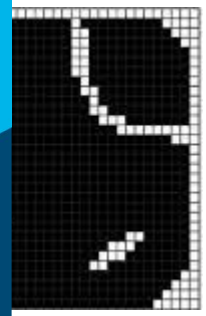
Boba-team:
Казовская Анастасия
Клименко Полина
Шульженко Александр



MAPF

MAPF (multi … а пути
для несколь…

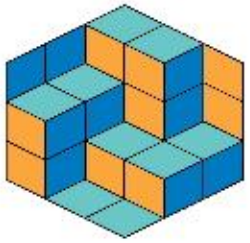
Инстанс зад…
каждого из …
стартовое и…



Задача: най…
чтобы агент…

Дополнител… …unction,
например, … …ентов.

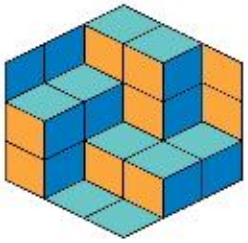
Применени…
робототехн…



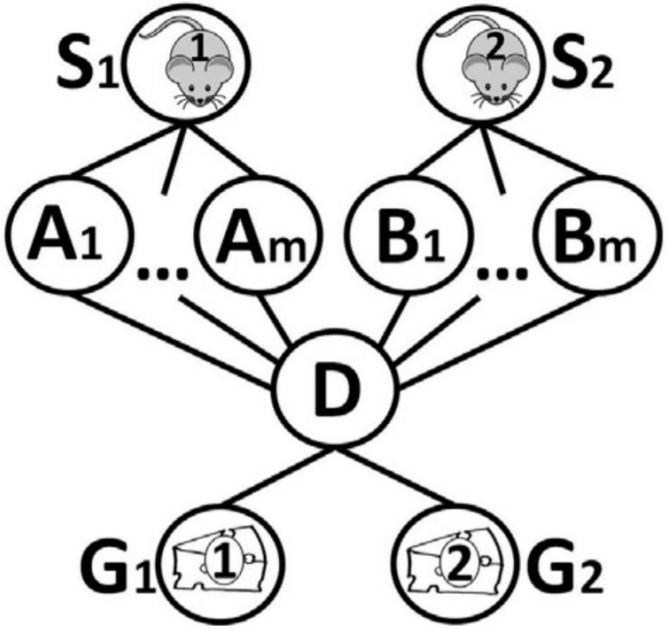
Формальная постановка задачи и определения

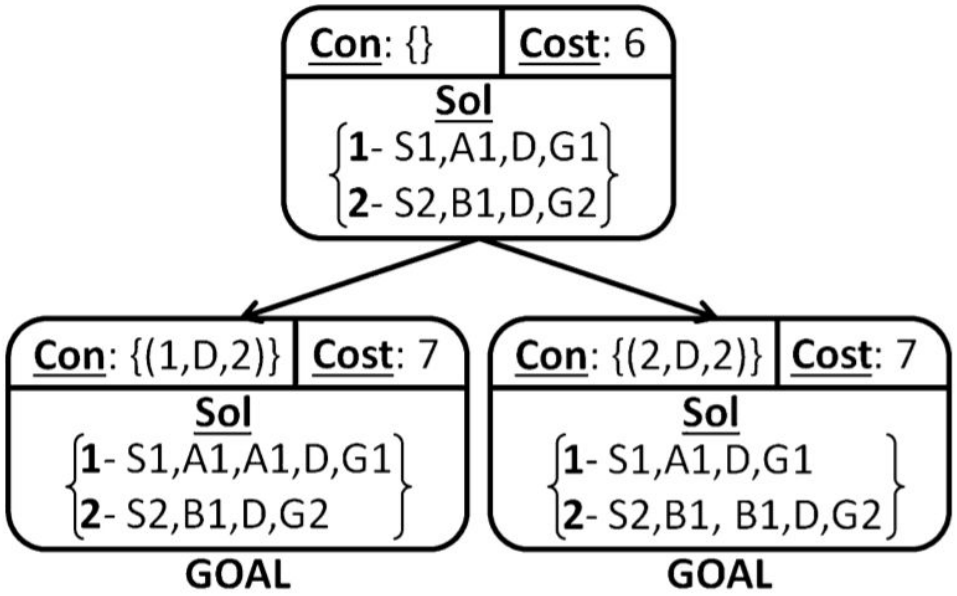
- Направленный граф G(V, E):
  - Вершины — возможные положения агентов
  - Ребра — допустимые перемещения между положениями

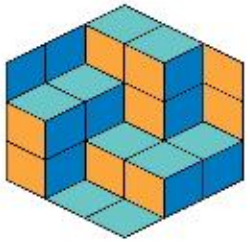
- k агентов: a_1, a_2, … , a_k, у каждого есть стартовое $start_i$ и целевое положение $goal_i$
- Время дискретно. В t_0 каждый агент a_i находится в $start_i$
- *Conflict* — (a_i, a_j, v, t). *Constraint* — (a_i, v, t)
- Cost-functions:
  - Sum-of-costs
  - Makespan
  - Fuel
  - Любая admissible cost-function



CBS. Пример


S1
1
2
S2
A1
...
Am
B1
...
Bm
D
G1
1
2
G2


Con: {}
Cost: 6
Sol
1- S1,A1,D,G1
2- S2,B1,D,G2
Con: {(1,D,2)}
Cost: 7
Sol
1- S1,A1,A1,D,G1
2- S2,B1,D,G2
GOAL
Con: {(2,D,2)}
Cost: 7
Sol
1- S1,A1,D,G1
2- S2,B1, B1,D,G2
GOAL

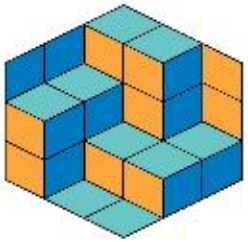


CBS. Псевдокод

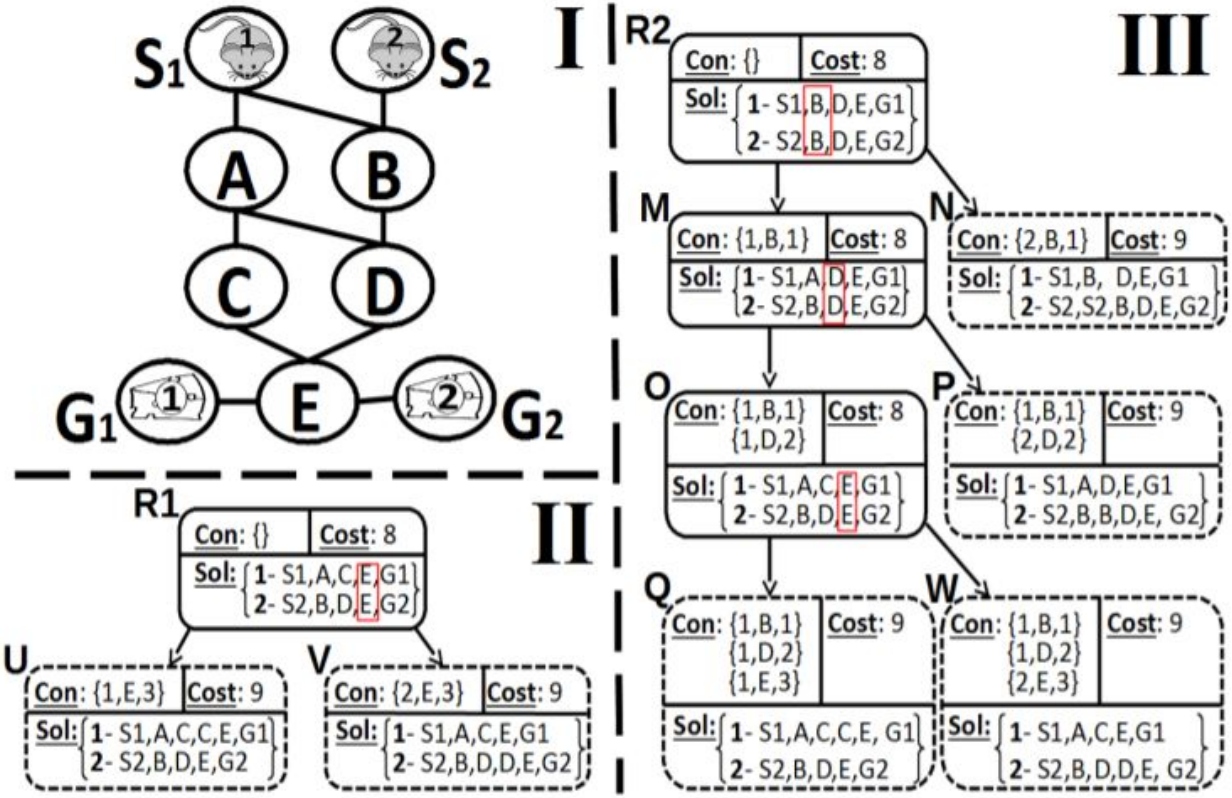
Algorithm 1: High-level of ICBS

```
1  Main(MAPF problem instance)
2      Init R with low-level paths for the individual agents
3      insert R into OPEN
4      while OPEN not empty do
5          N ← best node from OPEN // lowest solution cost
6          Simulate the paths in N and find all conflicts.
7          if N has no conflict then
8              return N.solution // N is goal
9          C ← find-cardinal/semi-cardinal-conflict(N) // (PC)
10         if C is not cardinal then
11             if Find-bypass(N, C) then // (BP)
12                 Continue
13         if should-merge(a_i, a_j) then // Optional, MA-CBS:
14             a_ij = merge(a_i,a_j)
15             if MR active then // (MR)
16                 Restart search
17             Update N.constraints()
18             Update N.solution by invoking low-level(a_ij)
19             Insert N back into OPEN
20             continue // go back to the while statement
21         foreach agent a_i in C do
22             A ← Generate Child(N, (a_i, s, t))
23             Insert A into OPEN
24 Generate Child(Node N, Constraint C = (a_i, s, t))
25     A.constraints ← N.constraints + (a_i, s, t)
26     A.solution ← N.solution
27     Update A.solution by invoking low level(a_i)
28     A.cost ← SIC(A.solution)
29     return A
```

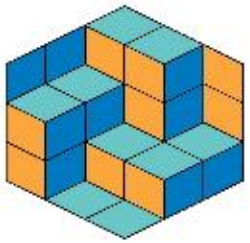
- **CBS**
- **Prioritize Conflicts**
- **Bypassing**
- **Meta-Agents**
- **Merge and Restart**



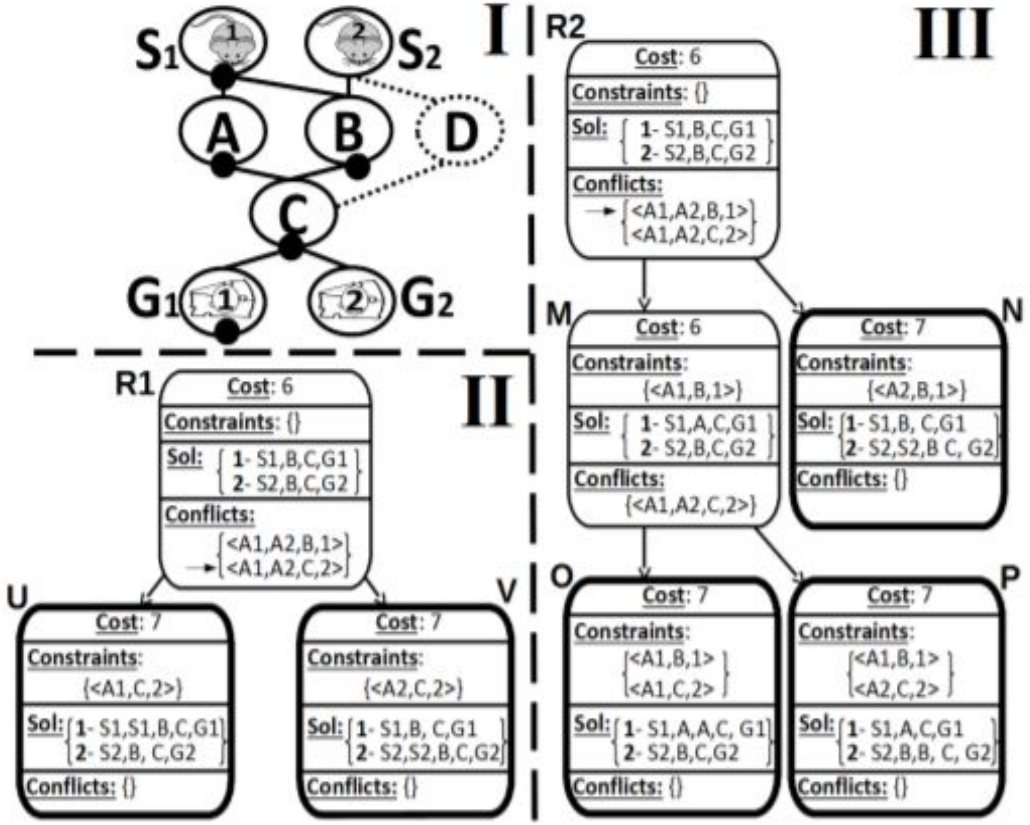
CBS. Bypassing



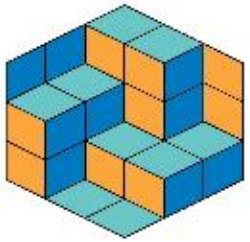
[CBS+BP. Псевдокод]



CBS. Prioritize Conflicts



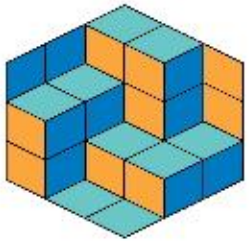
CBS+PC. Псевдокод



CBS. Meta-Agents

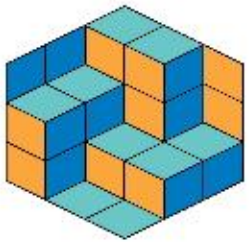
- MA-CBS — framework для любого MAPF-алгоритма
- Новая операция в Constraint Tree (CT): *merging* агентов в **мета-агента** при превышении лимита на число конфликтов
- Мета-агент никогда не разбивается на мета- или просто агентов
- Для “верхнего” уровня мета-агенты и просто агенты не отличимы
- “Нижний” уровень для мета-агентов использует оптимальный MAPF-алгоритм

MA-CBS. Псевдокод

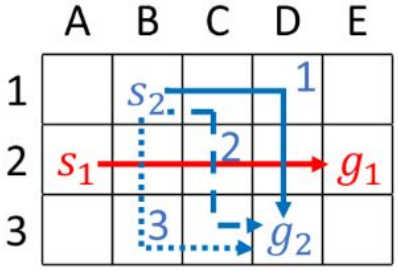


CBS. Disjoint Splitting

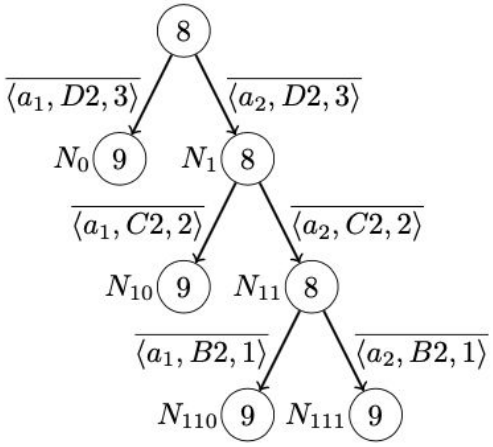
- На “верхнем” уровне возможно пересечение наборов допустимых траекторий движения агентов между вершинами CT
- Positive constraints — (a_i, v, t)
- Теперь для разных узлов CT наборы допустимых траекторий не пересекаются
- Ускорение в работе алгоритма



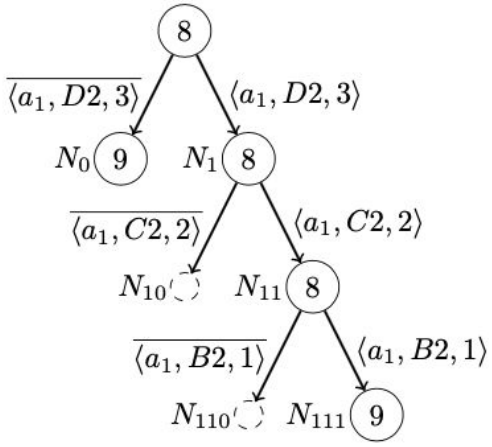
CBS. Disjoint Splitting. Пример



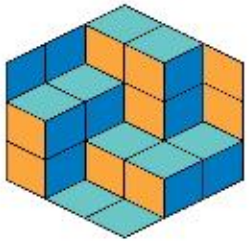
(a) A 2-agent MAPF instance



(b) Non-disjoint splitting



(c) Disjoint splitting

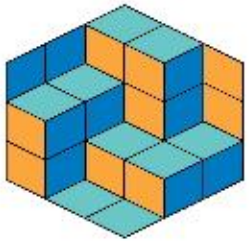


CBS. High-level heuristics

- CG:
  - Проверить, что в Node попал cardinal conflict
  - Построить cardinal conflict graph
  - Найти minimum vertex cover
  - h := size(MVC)

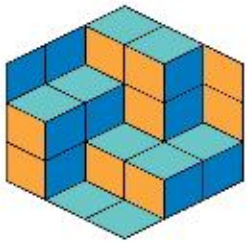
- DG:
  - Построить pairwise dependency graph
  - Найти minimum vertex cover
  - h := size(MVC)

- WDG:
  - Построить weighted pairwise dependency graph ($weight_{ij}$ = SOC_{ij}(Node) - min SOC_{ij})
  - Найти edge-weighted minimum vertex cover ($x_i + x_j \geqslant weight_{ij}$)
  - h := sum x_i



План экспериментов

- Виды карт:
  - City
  - Empty
  - Games
  - Mazes
  - Random
  - Rooms
  - Warehouse
- Baseline-алгоритмы:
  - A*
  - ICTS + pruning
  - Дополнительно: EPEA*
- Сравниваем:
  - Success rate от числа агентов (k)
  - Число порожденных вершин от k
  - Время работы от k (+ возможно, paired t-test)



План работ

- К 05 декабря:
  - CBS, MA-CBS, H — Настя
  - A*, ICTS + pruning — Полина
  - Подготовка материалов для тестирования (датасеты, функции для сбора статистики, рисования графиков и таблиц) — Саша
- К 15 декабря:
  - Тестирование CBS, MA-CBS, H — Настя
  - PC, BC, MR + тестирование — Полина
  - DS + тестирование — Саша
- К 23 декабря:
  - Доработка, исправление ошибок
  - Дополнительно: EPEA*, t-tests
  - Подготовка доклада