МЕТОДЫ И АЛГОРИТМЫ ЦЕНТРАЛИЗОВАННОГО ПЛАНИРОВАНИЯ СОВОКУПНОСТИ НЕКОНФЛИКТНЫХ ТРАЕКТОРИЙ ДЛЯ ГРУППЫ ИНТЕЛЛЕКТУАЛЬНЫХ АГЕНТОВ (МОБИЛЬНЫХ РОБОТОВ)

К.С. Яковлев
(1), (2)

А.А. Андрейчук
(3), (1)

(1) Федеральный исследовательский центр «Информатика и управление» Российской академии наук

(2) Высшая школа экономики

(3) Российский университет дружбы народов

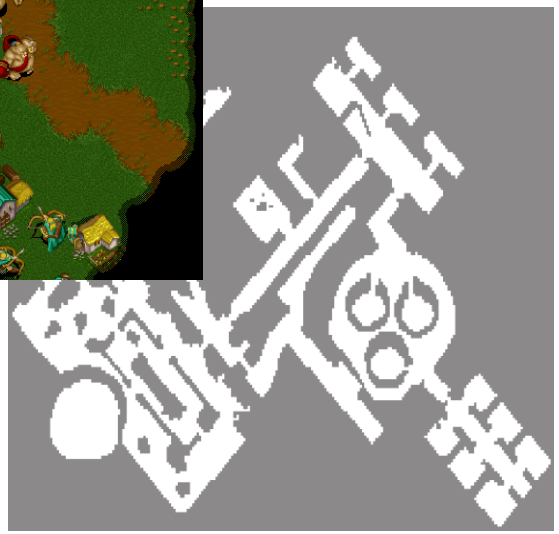
Мотивация

Реальные агенты:

- БПЛА
- Мобильные роботы

Виртуальные агенты:

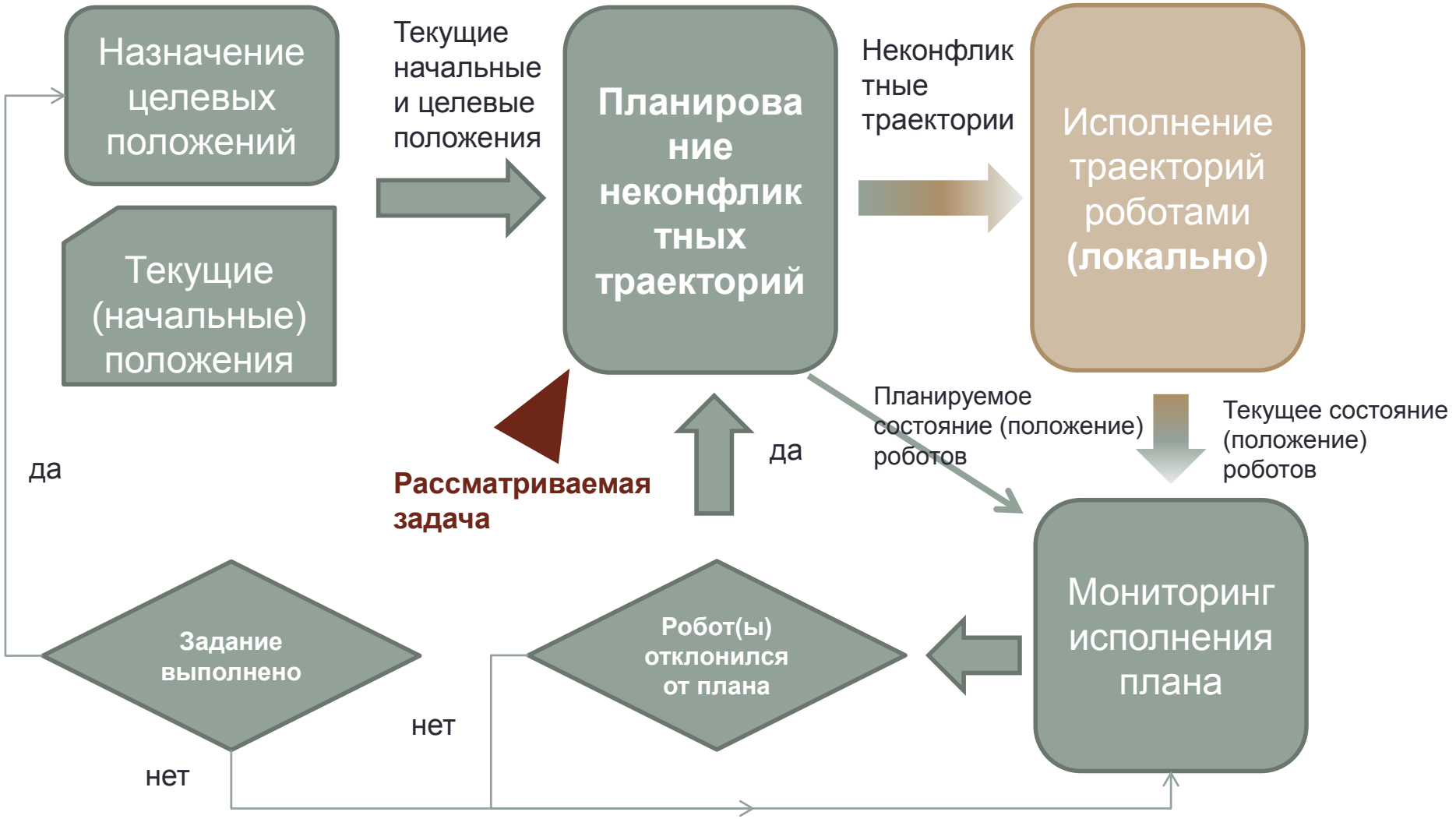
- Компьютерные игры



Сценарий «автоматизированный склад»

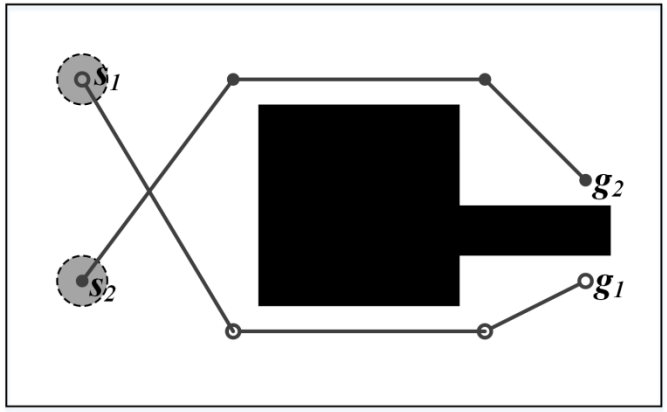
- Централизованный диспетчер и контроллер
- Внешняя система навигации высокой точности
- Гомогенная группа роботов
- Роботы точно исполняют план
  - Обеспечивается движение по траектории без отклонений

Обобщенный ход решения



Постановка задачи и допущения

1. Агент(ы) моделируется открытым диском радиуса ***r***
2. Агент(ы) может перемещаться в любом направлении (кинематические ограничения не учитываются)
3. Агент(ы) движется с постоянной скоростью **v**, может мгновенно останавливаться и набирать эту скорость (инерционные эффекты не учитываются)
4. Окружающая среда
   a. Полностью наблюдаема
   b. Содержит статические препятствия
5. Рабочая область (workspace) ***U*** – прямоугольник, состоящий из *Free Space* (проходимого пространства) и *Obstacles* (статических препятствий)
6. Состояние – точка на плоскости: **pos**=(x, y)
   a. Начальное и целевое состояние, **s** и **g**, фиксированы
7. Задана функция ***los***: $\boldsymbol{U} \times \boldsymbol{U} \rightarrow$ {true, false}, определяющая допустимость перехода агента из одного состояния в другое. ***los***($\mathbf{pos}_i$, $\mathbf{pos}_j$) = true $\Leftrightarrow$ *seg*($\mathbf{pos}_i$, $\mathbf{pos}_j$) $\cap$ *Obstacles* = $\varnothing$, где seg($\mathbf{p}_1$, $\mathbf{p}_2$) – отрезок прямой соединяющий $\mathbf{p}_1$ и $\mathbf{p}_2$.
8. **Путь**. $\pi(\mathbf{s}, \mathbf{g}) = (\mathbf{pos}_1, \mathbf{pos}_2, \ldots, \mathbf{pos}_m)$: ***los***($\mathbf{pos}_i$, $\mathbf{pos}_{i+1}$) = true $\forall i \in [1, m-1]$, $\mathbf{pos}_1 = \mathbf{s}$, $\mathbf{pos}_m = \mathbf{g}$.
9. **Траектория tr**(**s**, **g**)=[π(**s**, **g**), wait(m)], где wait(**s**, **g**)= ($wait_1$, $wait_2$, …., $wait_m$) – множество временных задержек, $wait_i \geq 0$



Задача

Дано: ***U***, ***los***, $\mathbf{s}_1$, $\mathbf{g}_1$ …, $\mathbf{s}_n$, $\mathbf{g}_n$

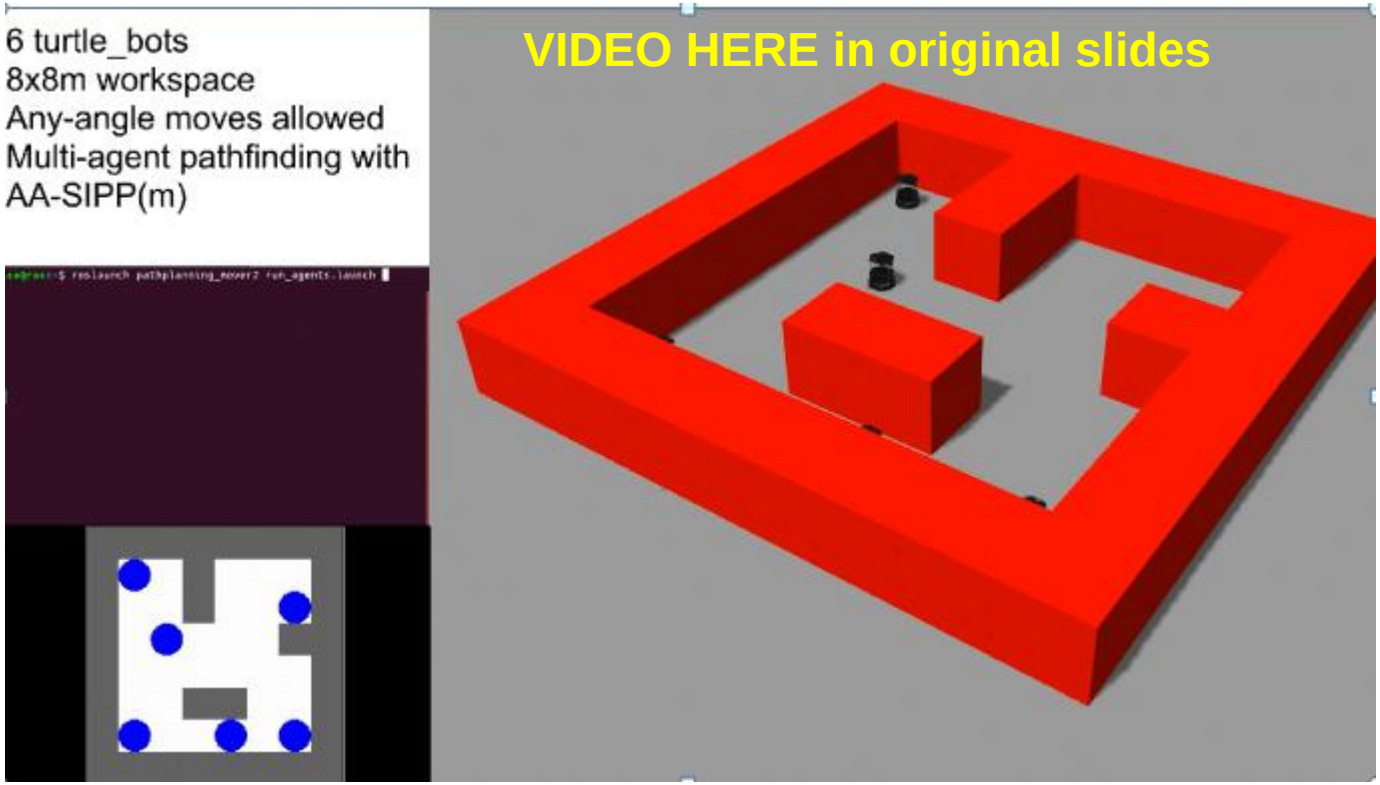
Найти: $\mathbf{tr}_1(\mathbf{s}_1, \mathbf{g}_1), \ldots, \mathbf{tr}_n(\mathbf{s}_n, \mathbf{g}_n)$ т.ч. при движении вдоль $\mathbf{tr}_i$ агент *i* избегает столкновений со статическими препятствиями и траекториями других агентов.

Критерий качества (чем меньше, тем лучше):

1. $\sum_{i=1}^{n} \boldsymbol{time}(\boldsymbol{tr}_i)$ – flowtime
2. $\max_{i=1..n}(\boldsymbol{time}(\boldsymbol{tr}_i))$ – makespan
3. $\sum_{i=1}^{n} \boldsymbol{length}(\boldsymbol{tr}_i)$
4. $\max_{i=1..n}(\boldsymbol{length}(\boldsymbol{tr}_i))$

Пример решения

- Симуляция в Gazebo (ROS + Gazebo)
- TurtleBots



Дискретизация

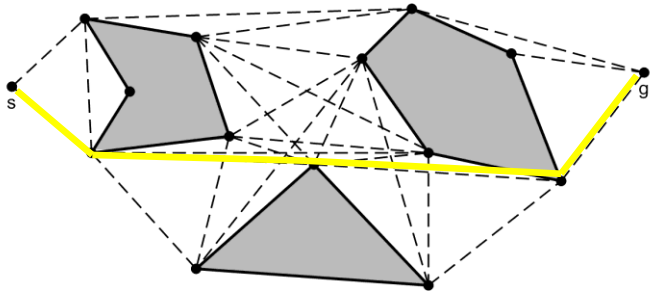
Планирование как поиск пути на графе особого вида

Вершины ~ положения агента в пространстве

Ребра ~ элементарные траектории (отрезки)

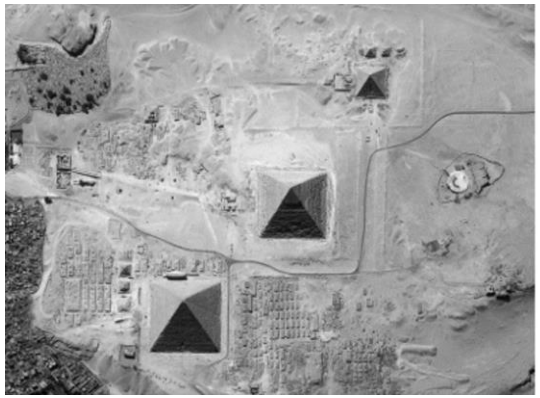
Веса ребер ~ длины соответствующих элемент. траекторий

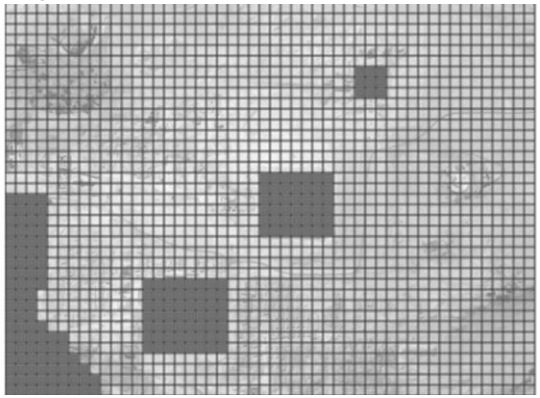
Путь на графе ~ траектория

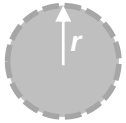


НЕОБХОДИМО

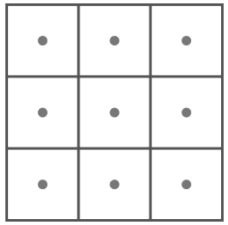
1. Построить граф
2. Найти *n* путей и рассчитать необходимые задержки при следовании по этим путям







Размер клетки: **2*r***



Положения агентов – центры клеток

Граф регулярной декомпозиции, MT-граф, grid - простая, информативная и легкая для построения модель, широко используемая в робототехнике

Подходы

1. Полное комплексирование (coupled planning)
   - Пространство поиска = декартово произведение n пространств
   - Поиск в этом пространстве известными алгоритмами (например – A*)
   - Учет независимых ветвей поиска: OD+ID
2. Комплексирование по мере необходимости
   - Conflict based search (CBS)
   - M*
3. Декомплексирование (decoupled planning)

3.1 Приоритезированное планирование
   - Назначение приоритетов + планирование с учетом движений высокоприоритезированных агентов

3.2 Планирование без учета движений других агентов
   - Устранение конфликтов после построения первоначального плана
   - Устранение конфликтов на этапе исполнения: MAPP, DiMPP

Ссылки

- **HCA*** (Silver, 2005)

  [Silver D. 2005. Cooperative pathfinding. In *Proceedings of The 1st Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE-2005)*, 117–122.]

- **OD+ID** (Standley, 2010)

  [Standley, T. S. 2010. Finding optimal solutions to cooperative pathfinding problems. In *Proceedings of The 24th AAAI Conference on Artificial Intelligence (AAAI-2010)*, 173–178.]

- **MAPP** (Wang and Botea, 2011)

  [Wang, K.-H.C., and Botea, A. 2011. MAPP: a scalable multi-agent path planning algorithm with tractability and completeness guarantees. *Journal of Artificial Intelligence Research*, (42):55–90.]

- **M*** (Wagner and Choset, 2011)

  [Wagner, G., and Choset, H. 2011. M*: A complete multirobot path planning algorithm with performance bounds. In *Proceedings of The 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS-2011)*, 3260-3267.]

- **CBS и его вариации**
  - CBS

    [Sharon, G., Stern, R., Felner, A., and Sturtevant, N.R. 2015. Conflict-based search for optimal multiagent path finding. *Artificial Intelligence Journal* (218):40–66.]

  - ECBS

    [Barer, M., Sharon, G., Stern, R. and Felner, A., 2014, July. Suboptimal variants of the conflict-based search algorithm for the multi-agent pathfinding problem. In *Proceedings of The 7th Annual Symposium on Combinatorial Search (SoCS-2014)*, 19–27.]

  - CBS + highways

    [Cohen, L., Uras, T., Kumar, T.S., Xu, H., Ayanian, N. and Koenig, S., 2016. Improved solvers for bounded-suboptimal multiagent path finding. In *Proceedings of The 25th International Joint Conference on Artificial Intelligence (IJCAI-2016)*, 3067-3074.]

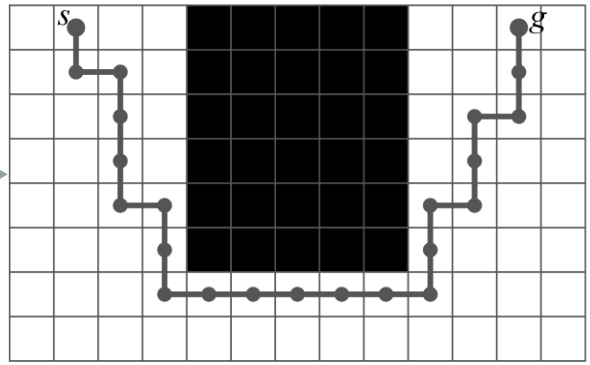
Перемещения в произвольном направлении

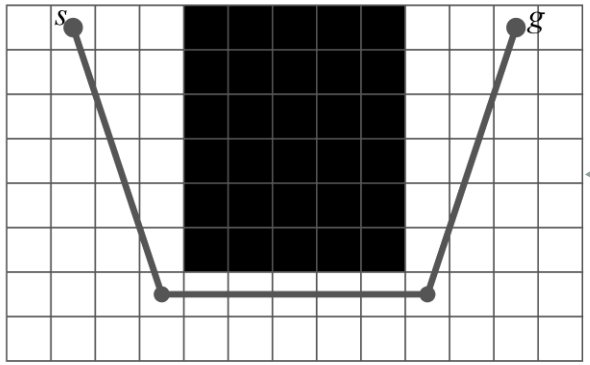
Традиционных подход:

- Агент может перемещаться только в **4** (или **8**) **смежные вершины**

Рассматриваемый подход:

- Агент может двигаться вдоль отрезка, который соединяет центры **любых** двух вершин (не обязательно смежных)





Допустимое движение – движение, при котором агент не задевает какое-либо препятствие (непроходимую клетку).

«Классическое» планирование: A*, JPS и др.

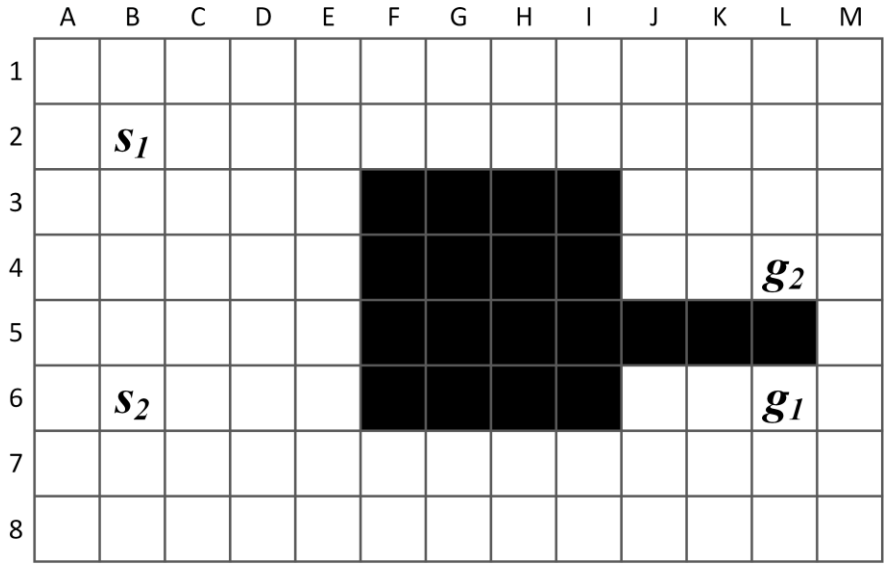
Any-angle планирование: Theta*, ANYA и др.

Конфликт в пространстве-времени (1/2)

Задан МТ-граф и заданы стартовые и целевые положения для двух агентов:

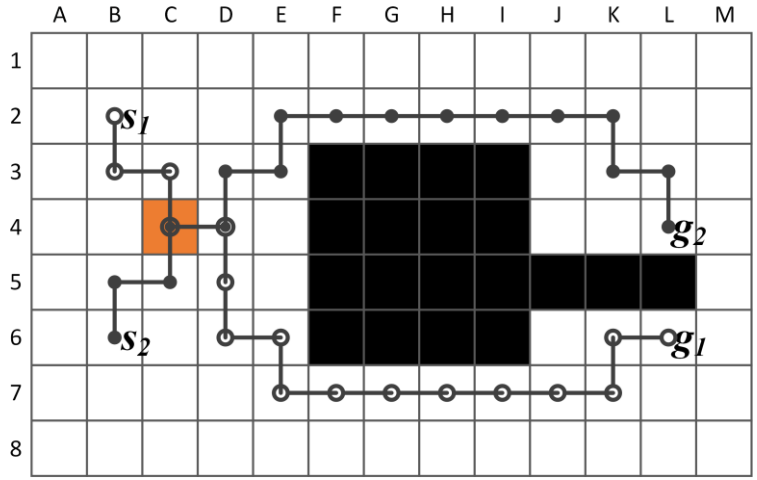
s_1=B2 g_1=L6

s_2=B6 g_2=L4



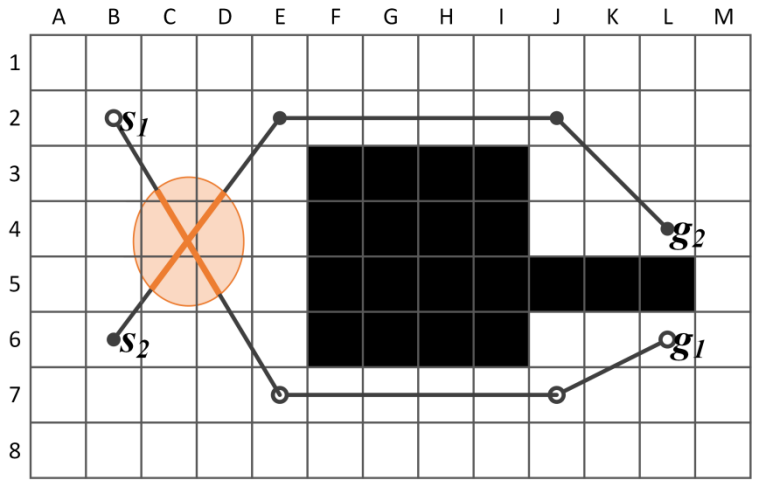
Конфликт в пространстве-времени (2/2)

Перемещения по горизонтали/вертикали



Легко определить (1) **где** и (2) **когда** возникает конфликт между агентами: конфликт возникает из-за того, что 2 агента пытаются занять вершину **C4** в момент времени **t = 3**.

Перемещения в произвольном направлении



(1) Конфликт между агентами возникает в некоторой области, а не в конкретной вершине или ребре MT-графа.
(2) Конфликт может возникнуть в любой момент времени (не дискретный).

Новый метод планирования и алгоритм его реализующий

AA-SIPP(m)

Yakovlev, K., Andreychuk, A. (2017) **Any-Angle Pathfinding for Multiple Agents Based on SIPP Algorithm**. In Proceedings of the 27th International Conference on Automated Planning and Scheduling (**ICAPS 2017**), Pittsburgh, PA, USA, June 18-23, 2017. pp. 586-593.

Предлагаемый подход

Приоритизированное планирование:

- Агентам присваиваются уникальные приоритеты.
- Агенты планируют траектории один за другим.
- Траектории агентов с более высоким приоритетом учитываются как динамические препятствия.

Поиск индивидуальных траектории алгоритмом **AA-SIPP**:

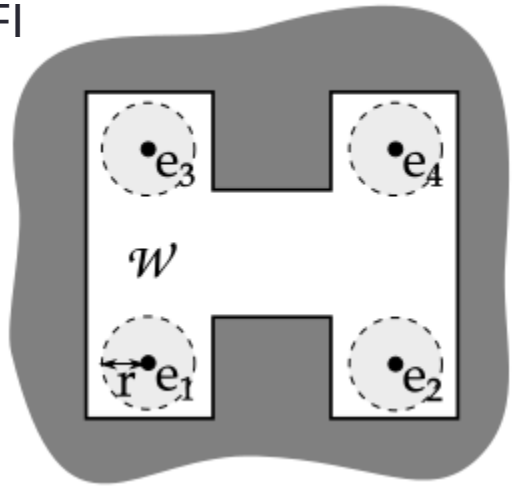
- AA-SIPP основан на предложенном ранее алгоритме SIPP.
- Может планировать в среде с динамическими препятствиями.
- **Агенты и динамические препятствия перемещаются в произвольном направлении.**

Результат – алгоритм **AA-SIPP(m)**

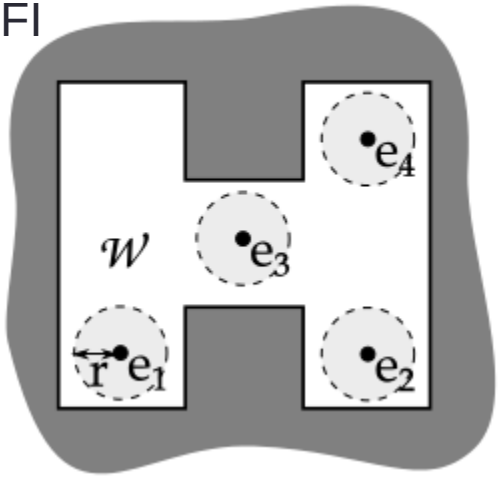
AA-SIPP(m)

- Минимизирует функционал качества, но не гарантирует достижение минимума.
- Гарантирует нахождение решения для специального класса задач – правильно-сформированных инфраструктур, w*ell-formed instrastructures* (в общем случае не является полным)
  - *WFI* – существуют все индивидуальные траектории, которые не пересекают стартовые и целевые положения других агентов.

WFI



non-WFI



Основа подхода

- Метод планирования в среде с динамическими препятствиями AA-SIPP
  - Динамические препятствия – высокоприоритезированные агенты, траектории для которых уже были найдены (и зафикисрованы)
- AA-SIPP = Safe Interval Path Planning + **Any-Angle**
  - **SIPP** [1] – «склеивание» отдельных временных точек (timepoints) в непрерывных временные интервалы и оперирование этими интервалами в процессе поиска

[1] Phillips, M., and Likhachev, M. 2011. SIPP: Safe interval path planning for dynamic environments. In *Proceedings of The 2011 IEEE International Conference on Robotics and Automation (ICRA-2011)*, 5628-5635.

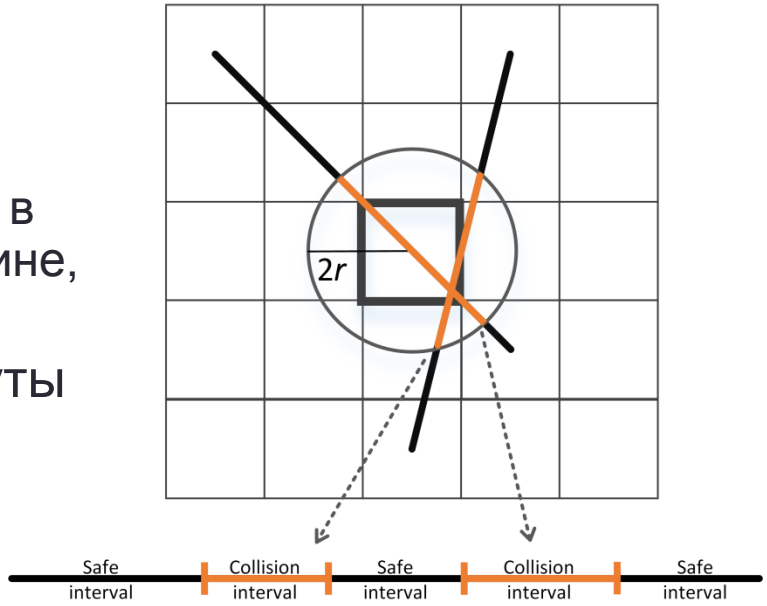
Алгоритм SIPP

Алгоритм SIPP осуществляет поиск в пространстве состояний $s = [cfg, interval]$:

- *cfg* – координаты вершины
- *interval* – безопасный интервал времени, в который агент может находиться в вершине, не создавая конфликтов

Также каждое состояние имеет атрибуты $g(s)$, $h(s)$, $parent(s)$, $time(s)$:

- *time* – наименьшее возможное время прибытия в вершину *cfg* (в течение безопасного интервала *interval*)
- *g* – стоимость лучшего найденного пути до *s* из стартового положения
- *h* – эвристическая оценка стоимости пути до целевого положения



Any-angle SIPP (AA-SIPP)

Algorithm 1: AA-SIPP

```
1  g(s_start) = 0; OPEN = ∅;
2  insert s_start into OPEN with f(s_start) = h(s_start);
3  while s_goal is not expanded do
4      remove s with the smallest f-value from OPEN;
5      for each cfg in NEIGHBORS(s.cfg) do
6          successors = getSuccessors(cfg, s);
7          if cfg is reachable from parent(s).cfg then
8              successors = successors ∪
                 getSuccessors(cfg, parent(s));
9          for each s' in successors do
10             if s' was not visited before then
11                 f(s') = g(s') = ∞;
12             if g(s') > g(s) + c(s, s') then
13                 g(s') = g(s) + c(s, s');
14                 updateTime(s');
15                 insert s' into OPEN with
                     f(s') = g(s') + h(s');
16 Function getSuccessors(cfg, s)
17     successors = ∅;
18     m_time = time to reach cfg from s.cfg;
19     start_t = time(s) + m_time;
20     end_t = endTime(interval(s)) + m_time;
21     intervals =get all safe intervals for cfg;
22     for each safe interval i in intervals do
23         if startTime(i) > end_t or
             endTime(i) < start_t then
24             continue;
25         t=earliest arrival time from s to cfg during
             interval i with no collisions;
26         if t does not exist then
27             continue;
28         s'=state of configuration cfg with interval i
             and time t;
29         insert s' into successors;
30     return successors;
```

AA-SIPP = SIPP + проверка на допустимость движения из родительского состояния - “shortcut”

Проверка на допустимость движения

Для проверки допустимости движения, необходимо найти все вершины, задеваемые агентом в процессе движения, и проверить их на проходимость.
Если хотя бы одна из вершин непроходима – движение недопустимо.

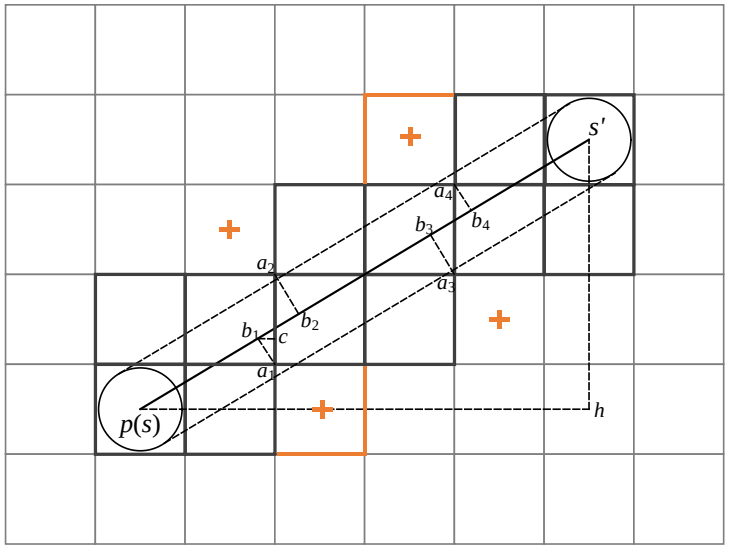
Алгоритм By

+

Дополнительные проверки

=

Алгоритм By+, который определяет все вершины, которые задевает агент.



Определение наличия конфликта

Конфликт между агентами существует, если:

$$|(\mathbf{pos} + \mathbf{v}t) - (\mathbf{pos'} + \mathbf{v'}t)| = r_{rob} + r_{obs},$$

т.е. существует момент времени, когда расстояние между агентом и динамическим препятствием меньше, чем сумма их радиусов.

Определив существование конфликта, добавляем задержку и повторяем проверку.

Свойства AA-SIPP, AA-SIPP(m)

AA-SIPP

Утверждение 1. Алгоритм является полным (относительно начальной топологии графа), т.е. гарантирует нахождение решения, если существует решение, состоящие из cardinal-only ребер.

Утверждение 2. Качество решений (flowtime), отыскиваемых алгоритмом AA-SIPP, не уступает качеству решений, отыскиваемых SIPP.

AA-SIPP(m) /начальные положения всех агентов считаются заблокированными/

Утверждение 1. Алгоритм является полным в классе задач well-formed infrastructure.

Экспериментальные исследования

Протестированные алгоритмы:

- **AA-SIPP(m) –** собственная реализация на C++.*
- **SIPP(m) –** собственная реализация на C++.*
- **ICBS** – реализация авторов алгоритма на C#.
- **ECBS** – реализация авторов алгоритма на C#.

Эксперименты проводились на ПК следующей конфигурации:

- ОС - Windows-8.1 x64
- CPU - AMD FX-8350(4.0 GHz)
- RAM - 16 Gb

*исходный код доступен на GitHub https://github.com/PathPlanning/AA-SIPP-m

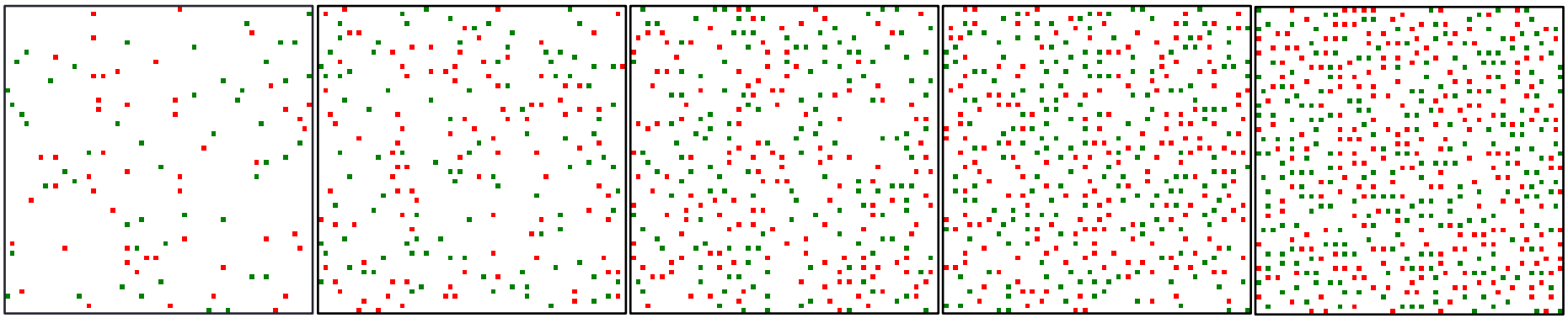
Эксперимент № 1. Входные данные

Карта – MT-граф размером 64 x 64 без препятствий

Задания:

- В каждом задании 50, 100, 150, 200 или 250 агентов
- 100 заданий на каждое число агентов
- Все задания гарантировано разрешимые (well-formed infrastructures)

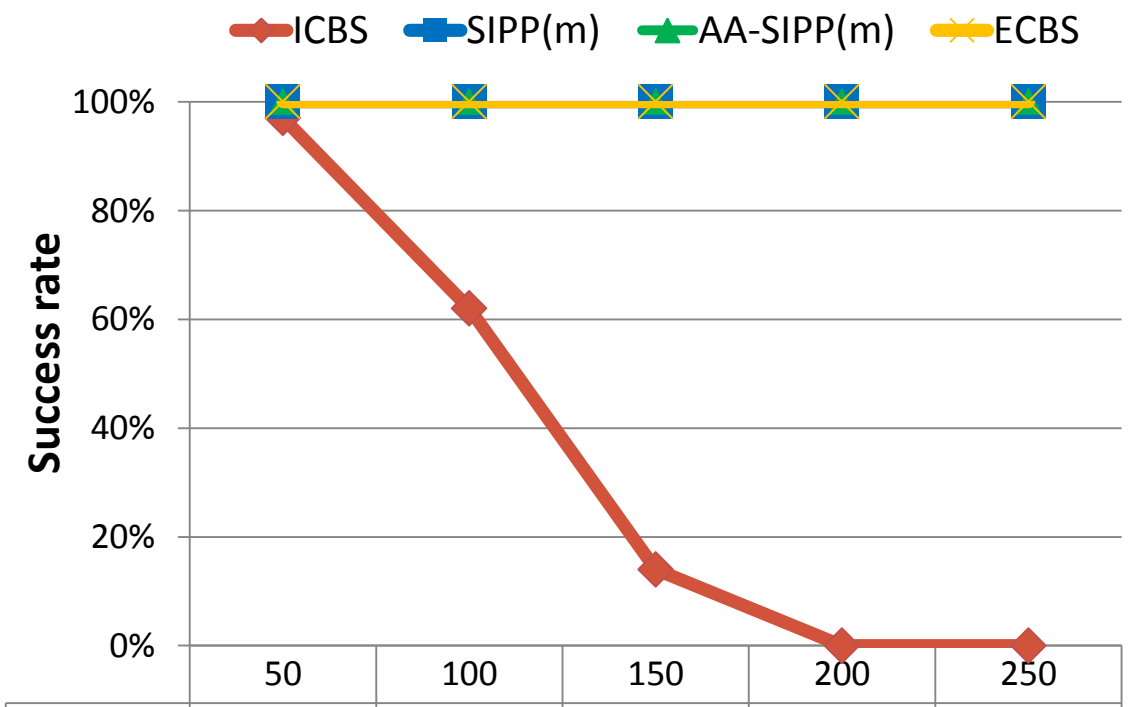
Время работы алгоритма ограничено 5 мин.



50 агентов 100 агентов 150 агентов 200 агентов 250 агентов

start goal

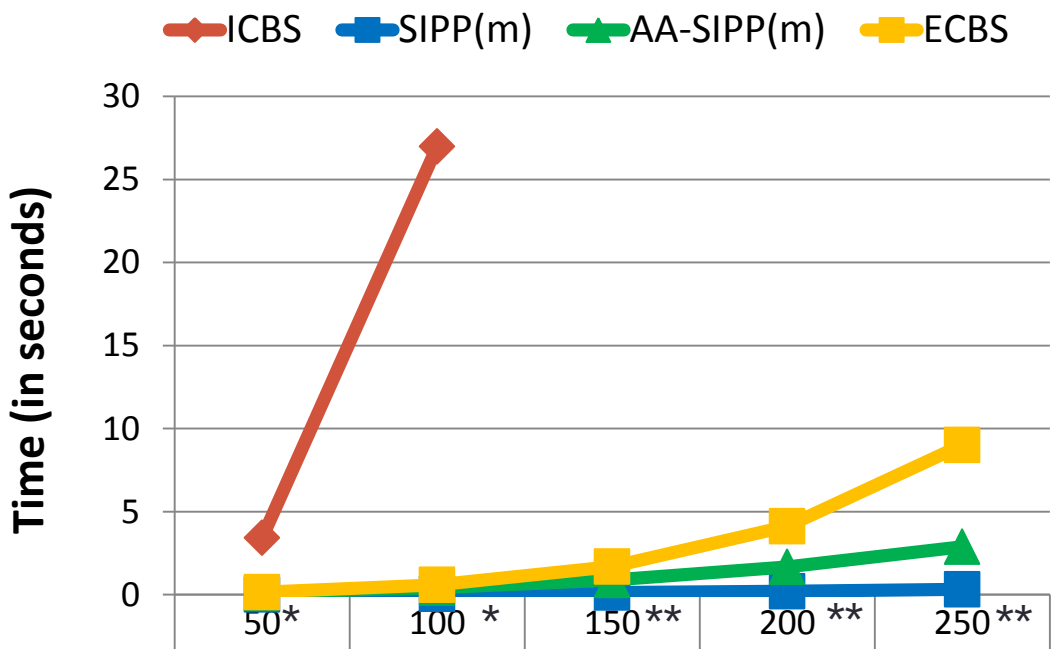
Эксперимент № 1. Процент успеха.



	50	100	150	200	250
ICBS	97%	62%	14%	0%	0%
SIPP(m)	100%	100%	100%	100%	100%
AA-SIPP(m)	100%	100%	100%	100%	100%
ECBS	100%	100%	100%	100%	100%

AA-SIPP(m):
Процент успеха - 100%
(т.к. задания класса well-formed infrastructure)

The 1st experiment. Runtime.



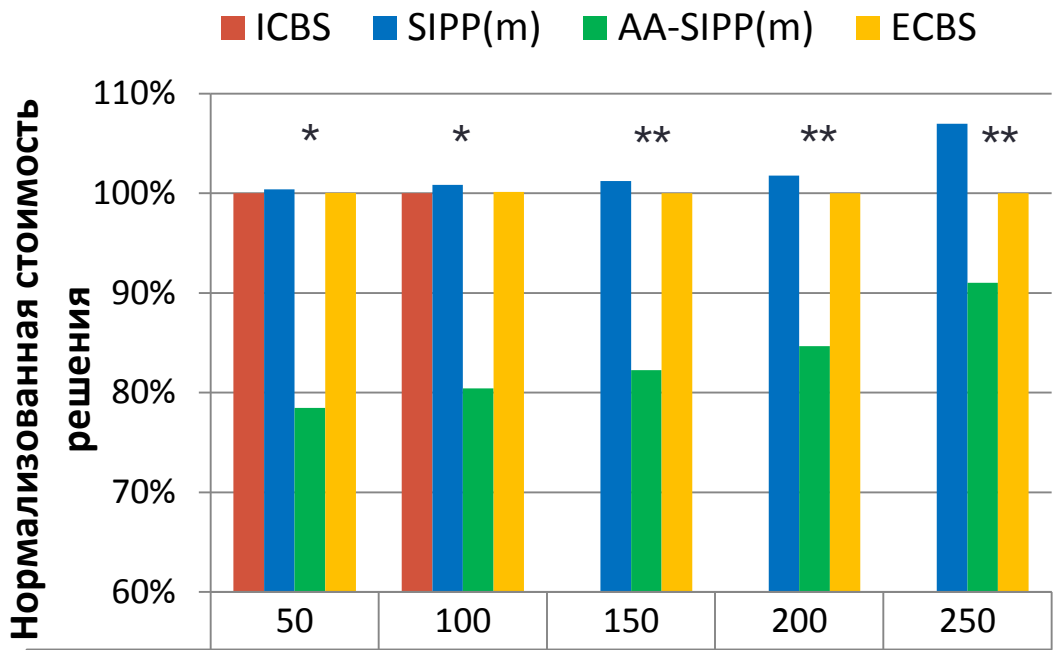
	50*	100 *	150**	200 **	250 **
ICBS	3,4176	26,9789	–	–	–
SIPP(m)	0,0155	0,0553	0,1236	0,2147	0,3344
AA-SIPP(m)	0,1311	0,4186	0,8931	1,6784	2,8774
ECBS	0,1421	0,5905	1,6989	4,1229	9,0114

AA-SIPP(m):
Хорошо масштабируется

*
Усреднение по всем заданиям, решенным ICBS

**
Усреднение по *всем* заданиям = заданиям, решенным SIPP(m), AA-SIPP(m), ECBS

Эксперимент № 1. Нормализованная стоимость решения



	50	100	150	200	250
ICBS	100,00%	100,00%	—	—	—
SIPP(m)	100,39%	100,85%	101,23%	101,77%	107,00%
AA-SIPP(m)	78,48%	80,44%	82,27%	84,66%	91,04%
ECBS	100,05%	100,15%	100,00%	100,00%	100,00%

AA-SIPP(m):
Стоимость решения ниже на 9-22%

*
100% = ICBS

**
100% = ECBS

Эксперимент № 1. Результаты

N		ICBS	SIPP(m)	AA-SIPP(m)	ECBS
50	Success	97.00%	100.00%	100.00%	100%
	Time(s)	3.4176	0.0155	0.1311	0.1421
	Cost	2240.56	2249.31 (+0.39%)	1758.29 (-21.52%)	2241.76 (+0.05%)
100	Success	62.00%	100.00%	100.00%	100%
	Time(s)	26.9789	0.0553	0.4186	0.5905
	Cost	4446.65	4483.66 (+0.83%)	3575.87 (-19.58%)	4452.28 (+0.13%)
150	Success	14.00%	100.00%	100.00%	100%
	Time(s)	–	0.1236	0.8931	1.6989
	Cost	–	6749.81	5485.56	6668.04
200	Success	0.00%	100.00%	100.00%	100%
	Time(s)	–	0.2147	1.6784	4.1229
	Cost	–	9106.2	7574.8	8947.8
250	Success	0.00%	100.00%	100.00%	100%
	Time(s)	–	0.3344	2.8774	9.0114
	Cost	–	11532.85	9812.7	10778.6

AA-SIPP(m):

- **Процент успеха – 100%**
- **Стоимость решения ниже на 9-22%**
- **Хорошо масштабируется**

Эксперимент № 2. Входные данные

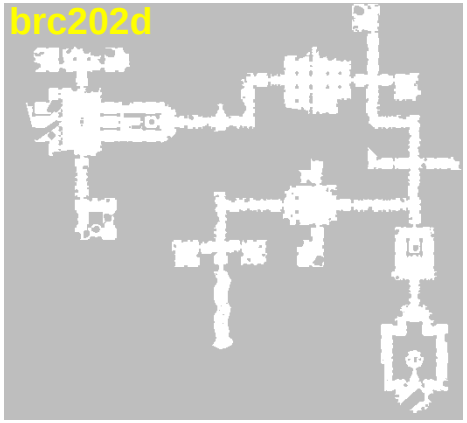
Карты - 3 карты “Dragon Age: Origins” из коллекции MovingAI benchmark.

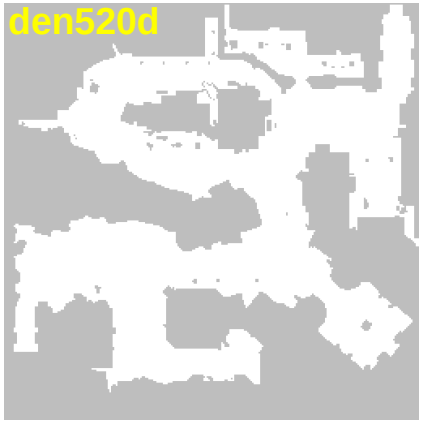
- Эти же карты использовались для тестирования в работах про ICBS.

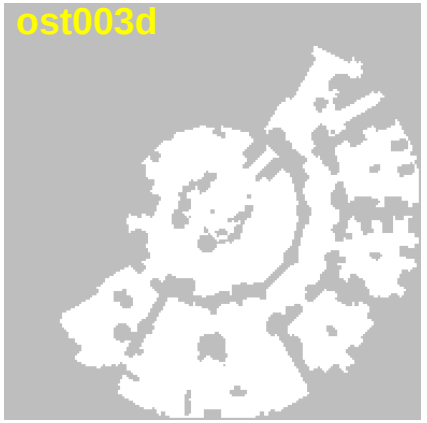
Задания:

- В каждом задании **25, 50, 70** или **100 агентов.**
- **100** заданий на каждую карту и число агентов (всего 1200).
- Задания не являются гарантированно разрешимыми для алгоритмов SIPP(m) и AA-SIPP(m).

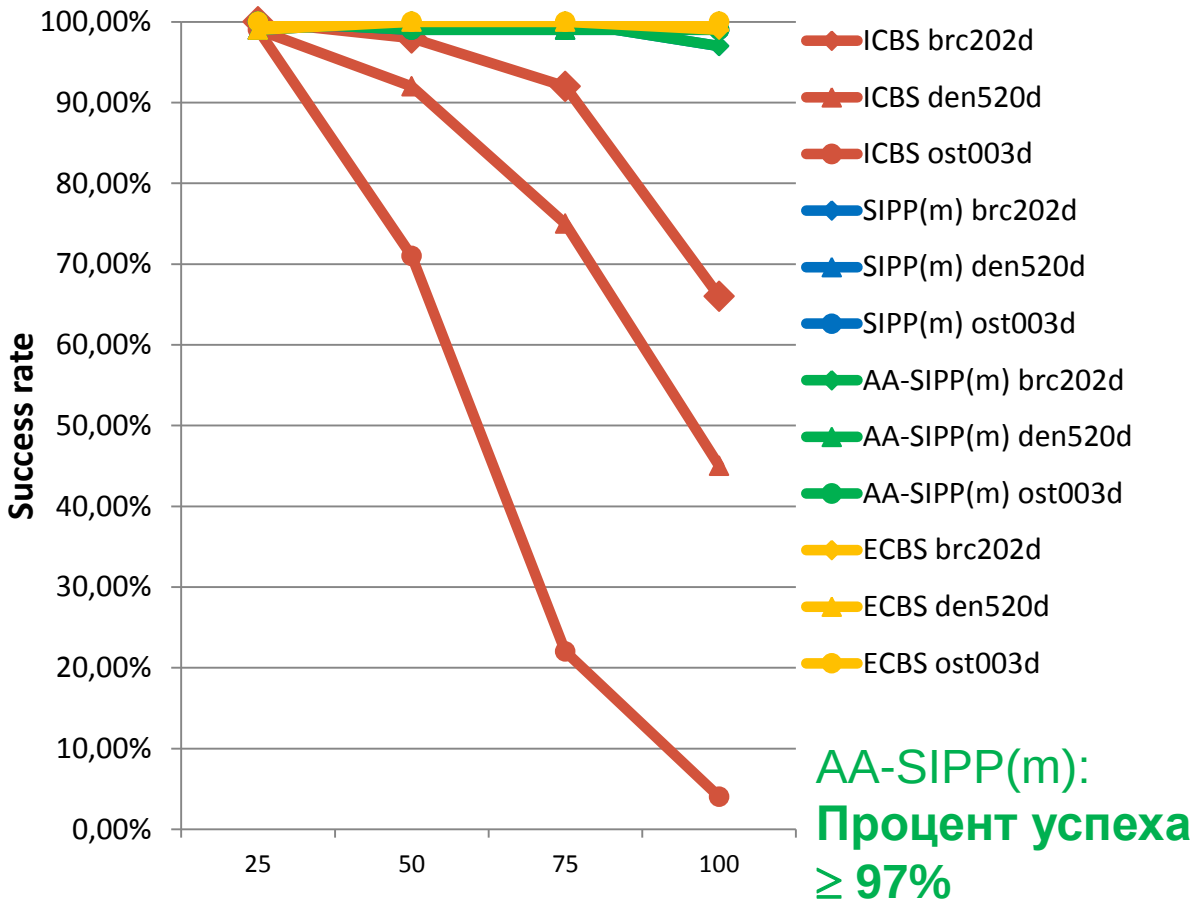
Время работы алгоритма ограничено 5 мин.





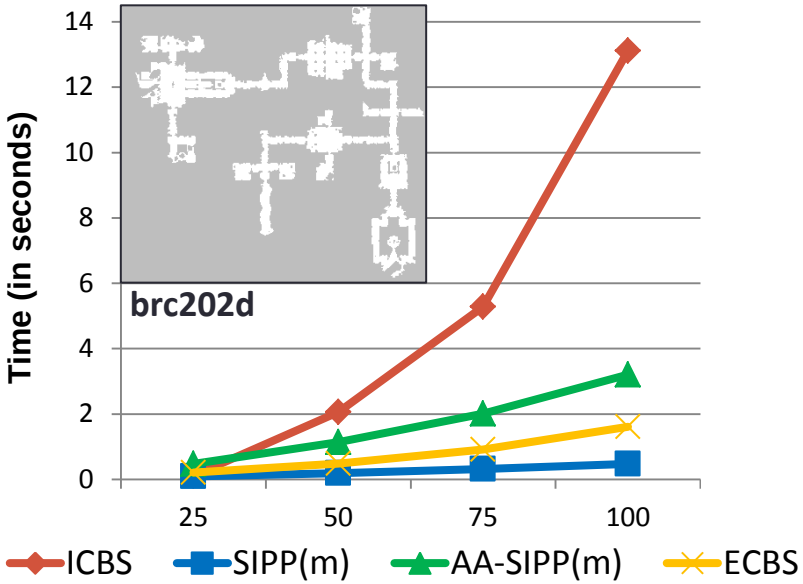


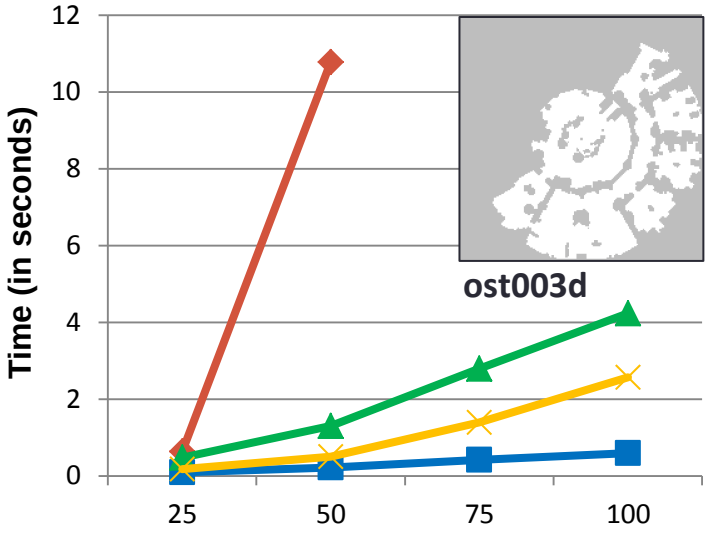
Эксперимент № 2. Процент успеха.

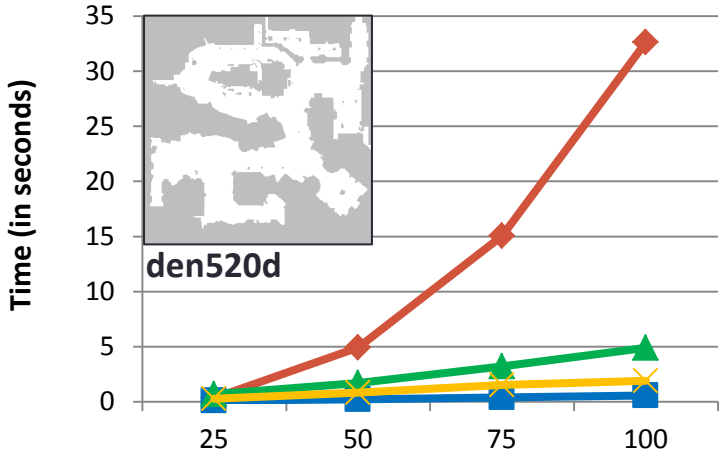


AA-SIPP(m):
Процент успеха ≥ 97%

Эксперимент № 2. Время.





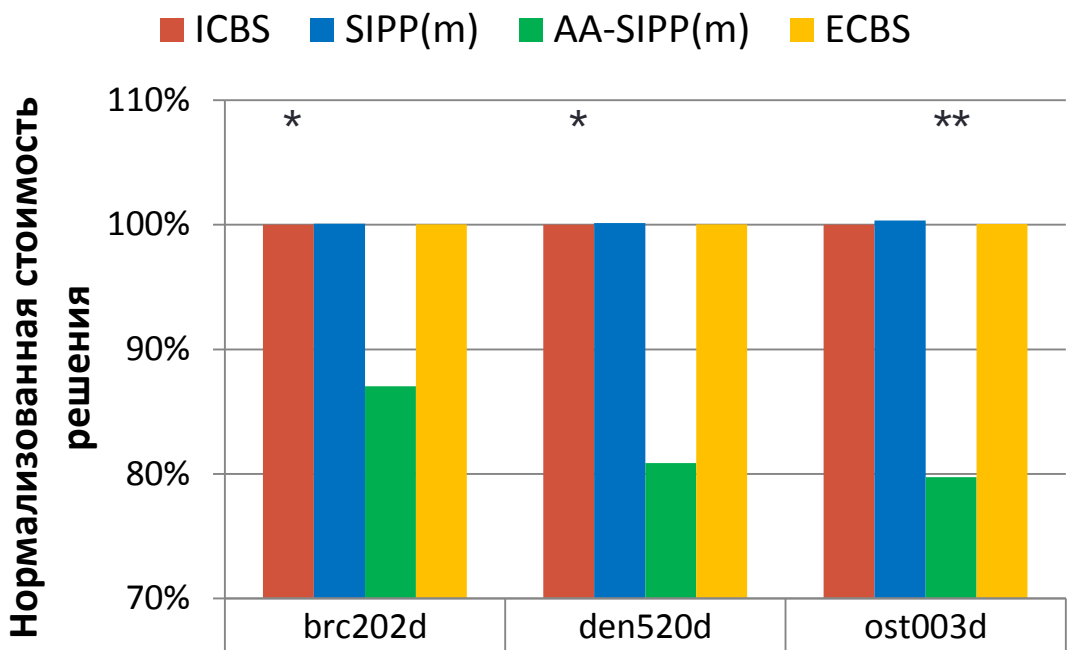


AA-SIPP(m):

Хорошо масштабируется

- но хуже, чем ECBS

Эксперимент № 2. Нормализованная стоимость решения



	brc202d	den520d	ost003d
ICBS	100%	100%	100%
SIPP(m)	100,09%	100,14%	100,35%
AA-SIPP(m)	87,03%	80,88%	79,74%
ECBS	100,02%	100,04%	100,07%

AA-SIPP(m):
Стоимость решения ниже на 13-21%

*
Усреднено по всем заданиям, которые решили все алгоритмы.

**
Усреднено по всем заданиям с 25 и 50 агентами, которые решили все алгоритмы.
(т.к. ICBS решил слишком мало заданий с 75 и 100 агентами на этой карте)

Эксперимент № 2. Результаты

		brc202d				den520d				ost003d			
N		ICBS	SIPP(m)	AA-SIPP(m)	ECBS	ICBS	SIPP(m)	AA-SIPP(m)	ECBS	ICBS	SIPP(m)	AA-SIPP(m)	ECBS
25	Success	100%	100%	100%	100%	99%	99%	99%	99%	99%	100%	100%	100%
	Time(s)	0.1105	0.0881	0.4735	0.2128	0.347	0.108	0.6873	0.2895	0.6365	0.0966	0.4814	0.1809
	Cost	3242.71	3244.13 (+0.04%)	2818.07 (-13.64%)	3243.12 (+0.01%)	3484.85	3487.18 (+0.07%)	2808.66 (-19.13%)	3485.8 (+0.03%)	2706.91	2711.43 (+0.17%)	2140.75 (-20.92%)	2708.12 (+0.04%)
50	Success	98%	100%	100%	100%	92%	100%	100%	100%	71%	99%	99%	100%
	Time(s)	2.056	0.1929	1.1353	0.4796	4.9273	0.2304	1.7087	0.8292	10.7776	0.2218	1.3025	0.5067
	Cost	6389.7	6394.75 (+0.08%)	5575.79 (-13.5%)	6390.99 (+0.02%)	6875.8	6884.33 (+0.12%)	5558.98 (-18.91%)	6877.93 (+0.03%)	5362.21	5380.37 (+0.34%)	4273.8 (-20.3%)	5367.3 (+0.09%)
75	Success	92%	100%	100%	100%	75%	99%	99%	100%	22%	99%	99%	100%
	Time(s)	5.2857	0.3128	2.0074	0.9185	15.0511	0.3865	3.2271	1.5346	–	0.4215	2.7905	1.3954
	Cost	9600.36	9611.16 (+0.11%)	8372.0 (-13.38%)	9603.35 (+0.03%)	10293.1	10311.4 (+0.18%)	8344.37 (-18.66%)	10298.2 (+0.05%)	–	8373.43	6663.58	8340.22
100	Success	66%	97%	97%	99%	45%	100%	100%	100%	4%	99%	99%	100%
	Time(s)	13.1123	0.4719	3.2037	1.6128	32.6364	0.5654	4.9068	1.9032	–	0.5931	4.2395	2.5662
	Cost	12806.2	12822.7 (+0.13%)	11094.1 (-13.37%)	12811.2 (+0.04%)	13665.3	13697.5 (+0.24%)	11105.4 (-18.73%)	13674.2 (+0.07%)	–	10959.5	8755.2	10906.4

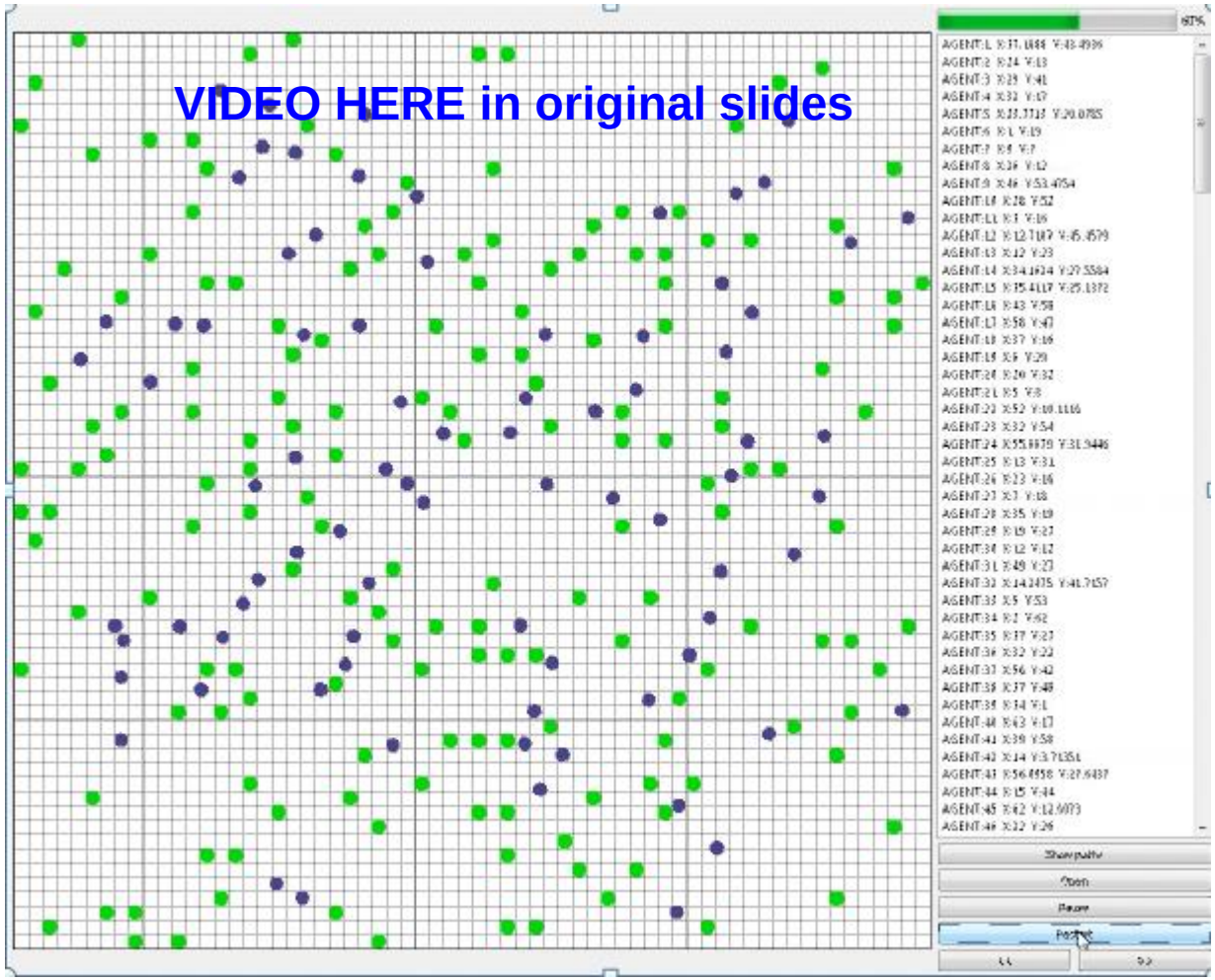
AA-SIPP(m):

- **Из всех 1200 протестированных заданий AA-SIPP(m) успешно решил 99,33%.**
- **Стоимость решения ниже на 13-21%.**
- **Хорошо масштабируется.**

Результаты

- Предложен алгоритм AA-SIPP, применимый для планирования траектории в среде с динамическими препятствиями и возможностью перемещения агента в произвольном направлении (any-angle planning).
- На основе алгоритма AA-SIPP предложен алгоритм планирования неконфликтных траекторий для группы агентов AA-SIPP(m).
- Проведенные экспериментальные исследования и сравнение с существующими алгоритмами показали высокую эффективность и преимущества алгоритма AA-SIPP(m).

Демонстрация (симуляция)



Методы и подходы к приоритизации агентов и модификации пространства поиска индивидуального планировщика

Anton Andreychuk and Konstantin Yakovlev. 2018. **Two Techniques That Enhance the Performance of Multi-robot Prioritized Path Planning**. In Proceedings of the 17th International Conference on Autonomous Agents and Multiagent Systems (**AAMAS 2018**), Stockholm, Sweden, July 10–15, 2018. (*in press*)

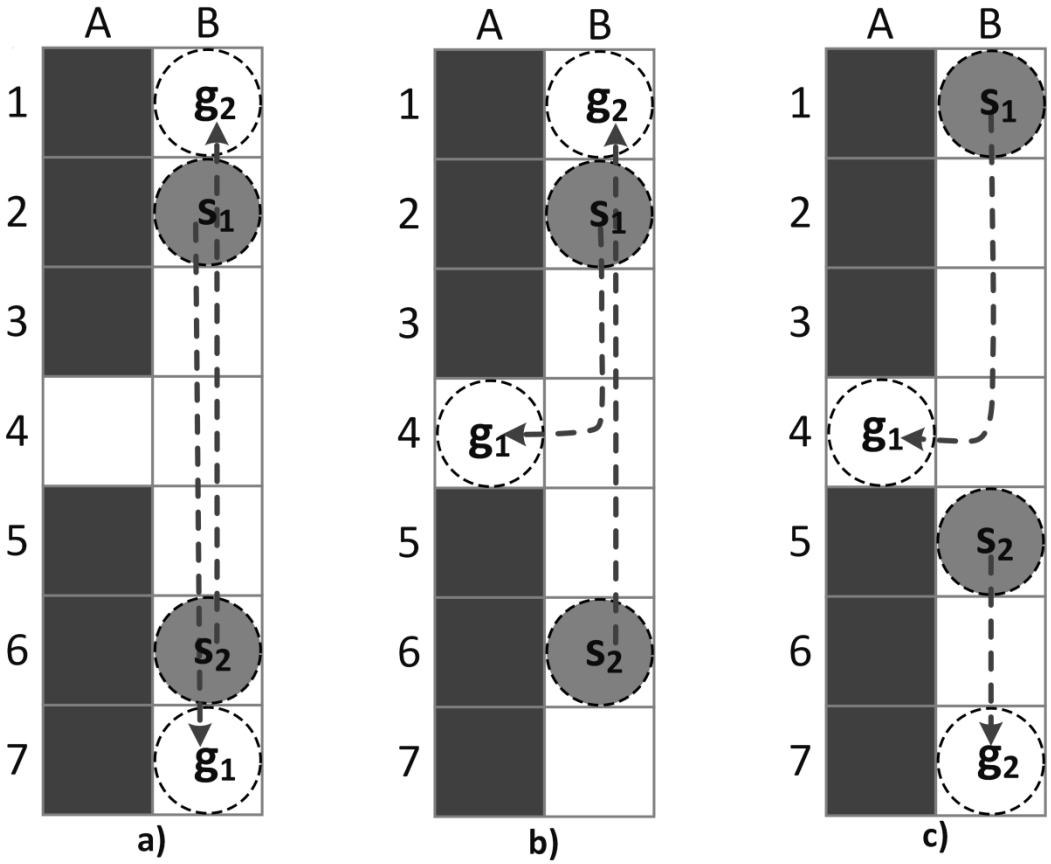
Приоритизация агентов

Существуют различные типы заданий:

а) Задание не разрешимо для приоритизированного алгоритма

б) Решение существует при определенной схеме приоритизации

в) Решение существует при любой схеме приоритизации

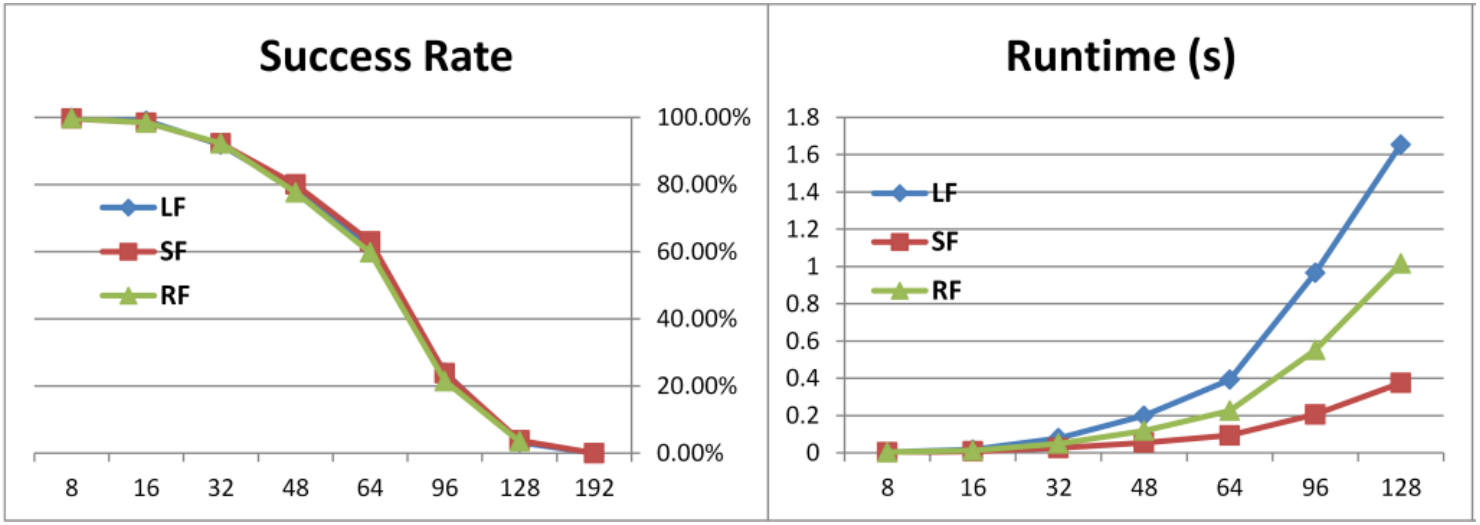


Влияние приоритизации на время работы алгоритма и success rate

SF(shortest first) – чем меньше расстояние между стартом и целью агента, тем выше его приоритет.

LF(longest first) – чем больше расстояние между стартом и целью агента, тем выше его приоритет.

RF(random first) – приоритет агентов назначен случайным образом

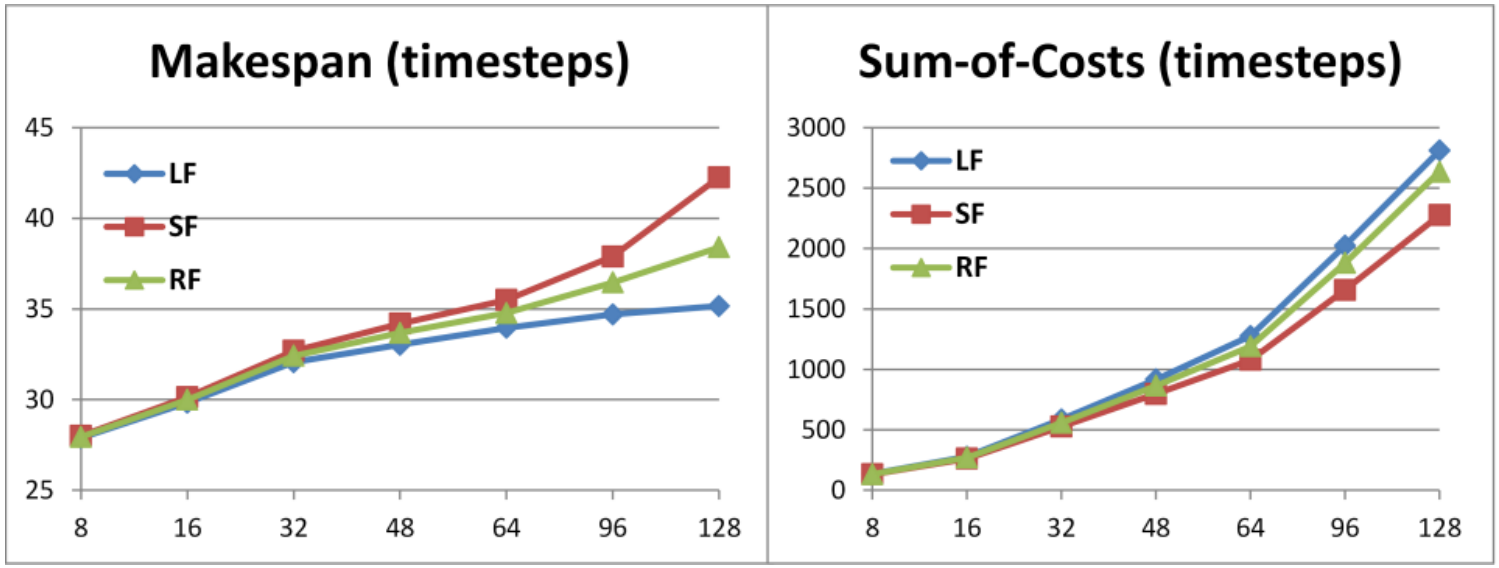


Влияние приоритизации на качество решения

SF(shortest first) – чем меньше расстояние между стартом и целью агента, тем выше его приоритет.

LF(longest first) – чем больше расстояние между стартом и целью агента, тем выше его приоритет.

RF(random first) – приоритет агентов назначен случайным образом

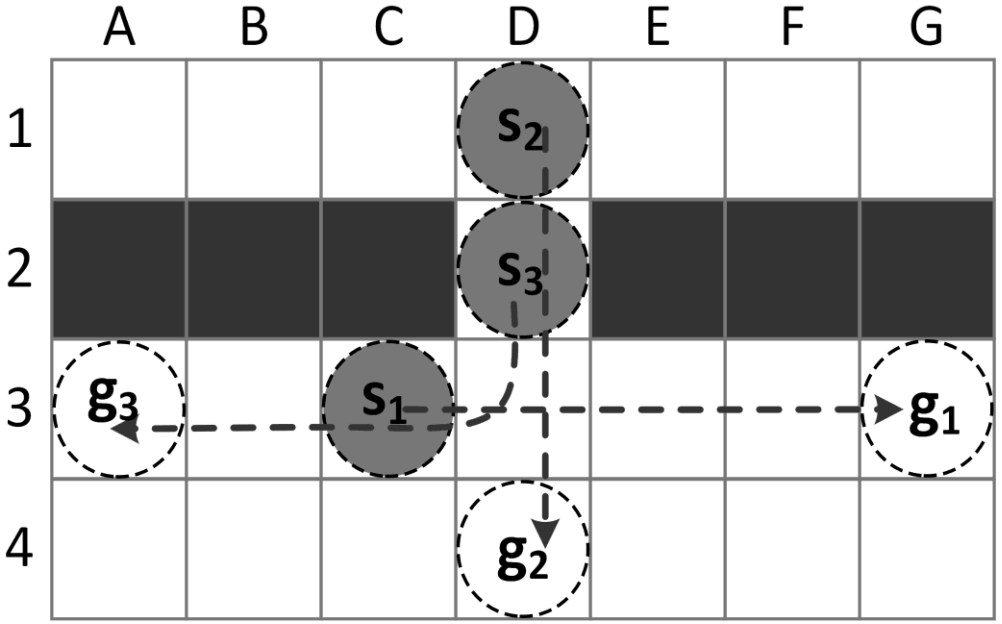


Промежуточный вывод

SF(shortest first) – наиболее предпочтительный способ, если только речь не идет о минимизации makespan «любой ценой»

Стартовые безопасные интервалы

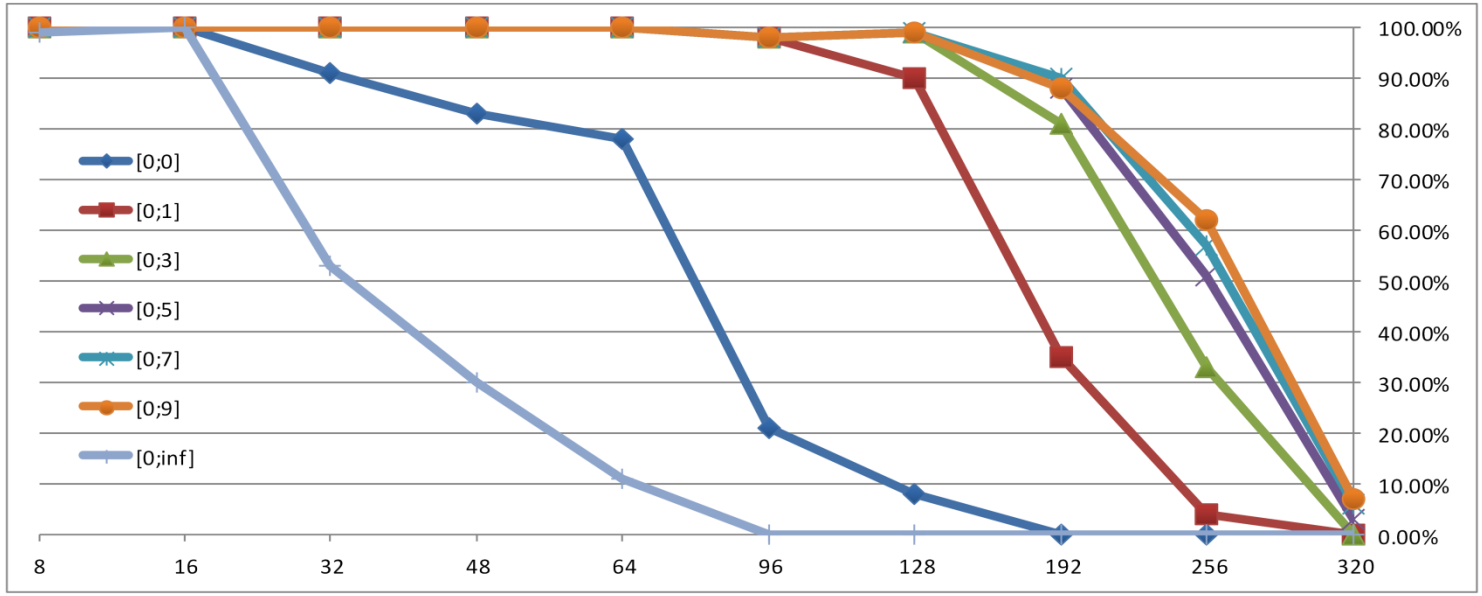
Проблема: решение не будет найдено, т.к. агент 3 должен подождать пока пройдет агент 1 и будет «сбит» агентом 2.



Возможное решение: добавление безопасных интервалов в стартовые вершины агентов, запрещающие другим агентам проходить через них определенное количество времени.

Экспериментальные исследования

- **[0;0] – стартовые вершины не заблокированы**
- **[0;n] – стартовые вершины заблокированы первые n шагов**
- **[0;inf] – стартовые вершины всегда заблокированы**



Добавление стартовых безопасных интервалов позволяет решать задания с гораздо большим числом агентов.

Перепланирование с изменением приоритетов агентов

В случаях, когда траектория одного из агентов не может быть найдена, изменение приоритетов может помочь решить задание.

Принцип работы предложенного подхода:

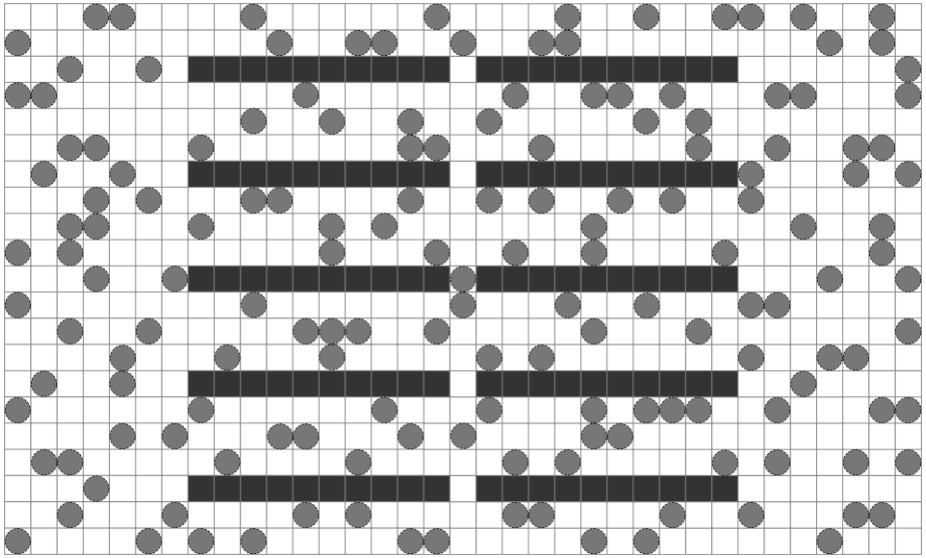
1) Повысить приоритет агенту, для которого не получилось построить траекторию. Таким образом, для этого агента гарантировано будет найдена траектория.
2) Если новая последовательность приоритетов уже была ранее – произошло зацикливание, решение не может быть найдено.
3) Попытаться найти решение с новой последовательностью приоритетов.

Algorithm 1: Prioritized planning with rule-based re-scheduling

```
1  PS = ∅; // set of priority sequences;
2  ChooseInitialPrioritySequence() ↦ ps = {i_1, i_2, ..., i_n};
3  while true do
4      TR = ∅; // set of agents' trajectories;
5      add ps to PS;
6      for j from 1 to n do
7          tr = FindTrajectoryOfTheIndividualAgent(i_j, TR);
8          if tr is found then
9              add tr to TR;
10             if j = n then
11                 return TR;
12                 //success! All the trajectories have been found
13         else
14             ps = {i_1 = i_j, i_2 = i_1, ..., i_n = i_{n-1}};
15             //raise the priority of agent i_j;
16             if ps ∈ PS then
17                 return failure;
18             break;
```

Экспериментальные исследования

Карта: MT-граф 21x35 моделирующий складские помещения.
Задания: по 100 заданий с 16, 32, 64, 96, 128, 160 и 192 агентами.
Время работы ограничено 5 мин.
Результаты:
Использование предложенного метода изменения приоритизации позволяет решать больше заданий с большим числом агентов и тратить меньше времени (за счет меньшего числа перепланирований)



	Random Restarts				Rule-Based Restarts			
	SR	t	Msp	SoC	SR	t	ms	SoC
16	1	0.01	33.31	285.15	1	0.01	33.31	285.15
32	1	0.04	36.68	591.3	1	0.04	36.68	591.3
64	1	0.22	41.25	1274.94	1	0.17	42.05	1252.82
96	1	0.68	50.05	2212.14	1	0.49	50.42	2149.31
128	1	5.73	62.23	3543.92	1	1.4	66.15	3322.52
160	0.42	117.82	79.73	5220.9	0.99	28.28	100.5	5298.85
192	0	—	—	—	0	—	—	—

Учет ориентации (heading) агента (робота) при планировании

AAt-SIPP(m)

Submitted to **IROS 2018** as “Improved Any-Angle Path Planner for Multi-Robot Navigation”

Экспериментальные исследования на реальных роботах – совместная работа с *Воробьевым В.В.*, НИЦ Курчатовский институт

AAt-SIPP

Учет направления движения агента приводит к:

А) Увеличению пространства поиска агента

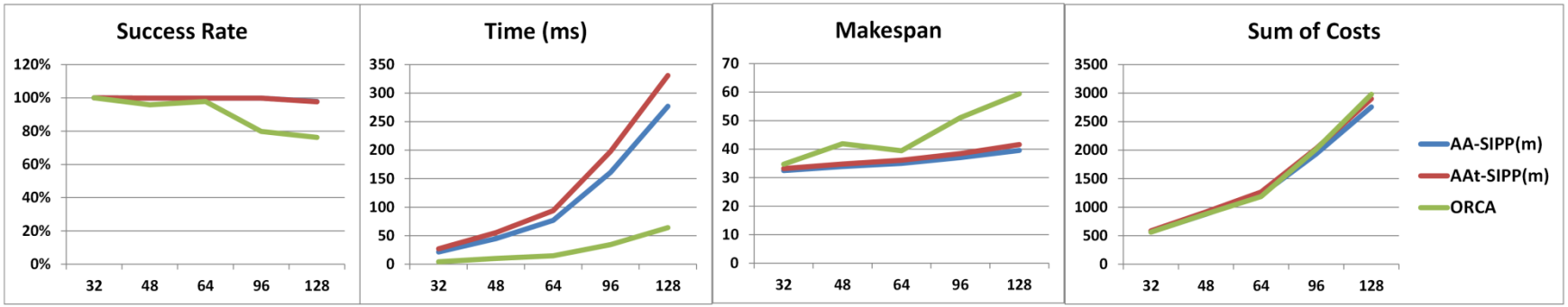
Б) Необходимости учета времени, требуемого для смены направления движения

Algorithm 1: AAt-SIPP (s_{start}, s_{goal})

```
1  g(s_start) = 0; OPEN = ∅;
2  insert s_start into OPEN with f(s_start) = h(s_start);
3  while s_goal is not expanded do
4      s := state with the smallest f-value in OPEN;
5      remove s from OPEN;
6      for each cfg in NEIGHBORS(s.cfg) do
7          successors := getSuccessors(cfg, s);
8          cfg' := cfg reachable from parent(s);
9          if cfg' exists then
10             successors =
                 successors ∪ getSuccessors(cfg', parent(s));
11         for each state s' in successors do
12             add_to_OPEN := true;
13             for each visited state s'' such that
                   s''.cfg.pos = s'.cfg.pos and
                   s''.interval = s'.interval do
14                 if g(s') ≥ g(s'')+dur_rot(s', s'') then
15                     add_to_OPEN := false;
16                 else if g(s'') > g(s') + dur_rot(s', s'') and
                       s'' ∈ OPEN then
17                     remove s'' from OPEN;
18             if add_to_OPEN = true then
19                 f(s') := g(s') + h(s');
20                 insert s' into OPEN;
```

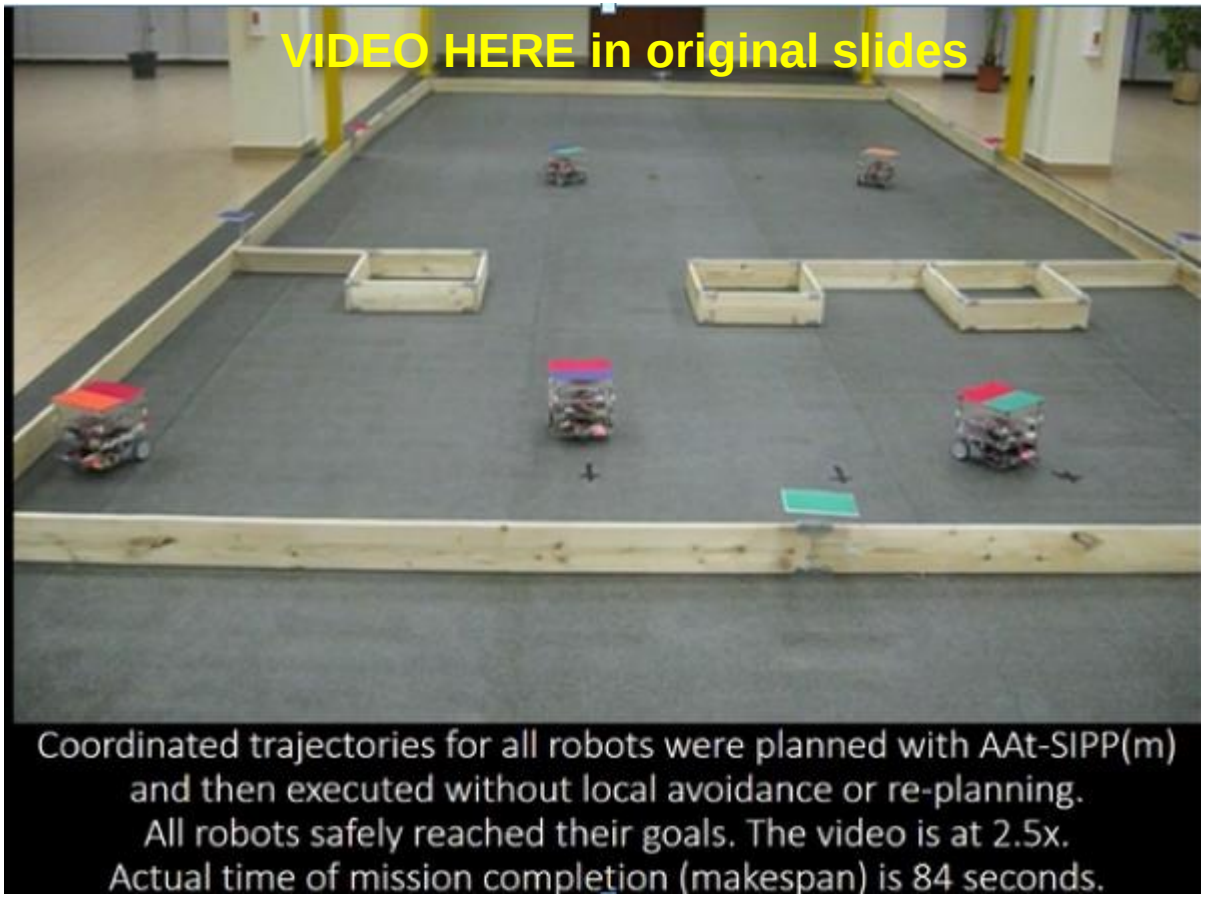
Экспериментальные исследования

Карта: MT-граф размером 32x32 без статических препятствий
Задания: по 500 заданий для 32, 48, 64, 96 и 128 агентов.
Задания сгенерированы случайным образом (не относятся к классу well-formed infrastructures).



Результаты:
А) учет направления движения агентов увеличивает время работы алгоритма на 20-25%.
Б) AA-SIPP(m) и AAt-SIPP(m) решают большее число заданий, чем ORCA

Демонстрация



Спасибо за внимание!

Статьи на arXiv
https://arxiv.org/a/yakovlev_k_1.html

Видео на youtube
Канал “Intelligent Robots”
https://www.youtube.com/channel/UChkz6VVLv9KJJg9XTcR0CLQ

Код на Github
https://github.com/PathPlanning/AA-SIPP-m

Контакты
yakovlev@isa.ru
kyakovlev@hse.ru