

# Многоагентные системы

Н. Д. Кудасов

МГУ им. Ломоносова

Москва, 2013

- 1 Концепция агента
- 2 Приложения МАС
- 3 Автономность в МАС
- 4 Разработка многоагентных систем
- 5 Список литературы

## Что такое агент?

“Агент — это инкапсулированная вычислительная система, помещенная в некоторую среду и способная автономно выполнять действия в этой среде для достижения поставленных целей.”

— Wooldridge and Jennings [WJ95]

“Агент — это вычислительная система, автономно действующая от лица других сущностей, выполняющая действия реактивно и/или с определенной целью и в некоторой степени использующая свойства обучаемости, кооперативности и мобильности.”

— Shaw Green et al. [GHN<sup>+</sup>97]

В самом общем случае агент обладает следующими свойствами:

- *реактивность*: способность агента воспринимать окружающее и влиять на него;
- *целеустремленность*: агент должен действовать в заложенными в него целями;
- *социальная активность*: агент должен взаимодействовать с другими агентами и/или людьми;
- *автономность*: агент действует без непосредственного вмешательства человека и обладает определенным контролем на своими действиями и внутренним состоянием.

В самом общем случае агент обладает следующими свойствами:

- *реактивность*: способность агента воспринимать окружающее и влиять на него;
- *целеустремленность*: агент должен действовать в заложенными в него целями;
- *социальная активность*: агент должен взаимодействовать с другими агентами и/или людьми;
- *автономность*: агент действует без непосредственного вмешательства человека и обладает определенным контролем на своими действиями и внутренним состоянием.

В самом общем случае агент обладает следующими свойствами:

- *реактивность*: способность агента воспринимать окружающее и влиять на него;
- *целеустремленность*: агент должен действовать в заложенными в него целями;
- *социальная активность*: агент должен взаимодействовать с другими агентами и/или людьми;
- *автономность*: агент действует без непосредственного вмешательства человека и обладает определенным контролем на своими действиями и внутренним состоянием.

В самом общем случае агент обладает следующими свойствами:

- *реактивность*: способность агента воспринимать окружающее и влиять на него;
- *целеустремленность*: агент должен действовать в заложенными в него целями;
- *социальная активность*: агент должен взаимодействовать с другими агентами и/или людьми;
- *автономность*: агент действует без непосредственного вмешательства человека и обладает определенным контролем на своими действиями и внутренним состоянием.

# Ментальные характеристики

Распространено использование ментальных характеристик:

- знания,
- убеждения,
- намерения,
- обязательства и т.п.

Иногда агенты наделяются *эмоциями*.



Часто также используются следующие свойства агентов:

- *мобильность*: способность агентов перемещаться (физически или в сети);
- *правдивость*: предположение, что агент не может намеренно фальсифицировать передаваемую (другим агентам, человеку) информацию;
- *доброжелательность*: предположение, что цели агентов не конфликтуют и, следовательно, каждый агент стремится выполнить то, о чём его просят;
- *рациональность*: предположение, что агент действует в соответствии со своими целями и не пытается противостоять себе (по крайней мере, насколько это позволяют его убеждения).

Часто также используются следующие свойства агентов:

- *мобильность*: способность агентов перемещаться (физически или в сети);
- *правдивость*: предположение, что агент не может намеренно фальсифицировать передаваемую (другим агентам, человеку) информацию;
- *доброжелательность*: предположение, что цели агентов не конфликтуют и, следовательно, каждый агент стремится выполнить то, о чём его просят;
- *рациональность*: предположение, что агент действует в соответствии со своими целями и не пытается противостоять себе (по крайней мере, насколько это позволяют его убеждения).

Часто также используются следующие свойства агентов:

- *мобильность*: способность агентов перемещаться (физически или в сети);
- *правдивость*: предположение, что агент не может намеренно фальсифицировать передаваемую (другим агентам, человеку) информацию;
- *доброжелательность*: предположение, что цели агентов не конфликтуют и, следовательно, каждый агент стремится выполнить то, о чём его просят;
- *рациональность*: предположение, что агент действует в соответствии со своими целями и не пытается противостоять себе (по крайней мере, насколько это позволяют его убеждения).

Часто также используются следующие свойства агентов:

- *мобильность*: способность агентов перемещаться (физически или в сети);
- *правдивость*: предположение, что агент не может намеренно фальсифицировать передаваемую (другим агентам, человеку) информацию;
- *доброжелательность*: предположение, что цели агентов не конфликтуют и, следовательно, каждый агент стремится выполнить то, о чём его просят;
- *рациональность*: предположение, что агент действует в соответствии со своими целями и не пытается противостоять себе (по крайней мере, насколько это позволяют его убеждения).

Существует ряд теорий и концепций, использующихся для описания агентов:

- *логические системы*: цели и свойства агента описываются при помощи высказываний в различных логических системах;
- *системы намерений*: внутреннее состояние агента представляется системой мировоззрений (знания, убеждения, желания, намерения и т.п.);
- *модели коммуникаций*: взаимодействие между агентами происходит посредством специальных действий.

Многоагентная система (МАС) — это система из нескольких взаимодействующих интеллектуальных агентов.

МАС используются для:

- регулирования траффика;
- онлайн-торговли;
- моделирования соц. структур;
- группового ИИ в играх и фильмах;
- устранения чрезвычайных ситуаций (ЧС);
- сенсорных сетей и т.д.

- 1 Концепция агента
- 2 Приложения МАС**
- 3 Автономность в МАС
- 4 Разработка многоагентных систем
- 5 Список литературы

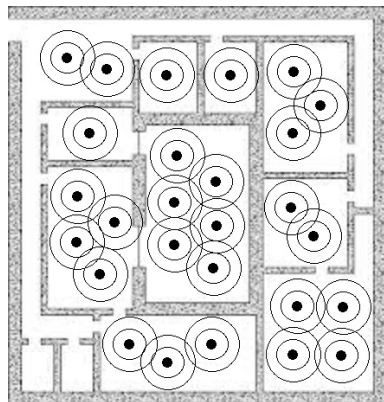
# Сенсорная сеть

Сенсорная сеть [RK07] состоит из автономных сенсоров, постоянно собирающих информацию о температуре/влажности/давлении и пр. Для передачи информации соседние сенсоры должны иметь разные частоты.

Например:

- каждый сенсор может выбрать 1 из 3 частот;
- соседние сенсоры не должны иметь одну частоту.

Эта задача эквивалентна задаче о раскраске графа в 3 цвета.



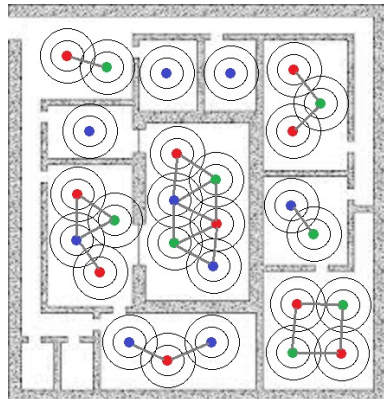


# Раскраска графа

Условия:

- каждая вершина может быть раскрашена в один из трех цветов;
- смежные вершины должны быть раскрашены в разные цвета.

Обе задачи являются примером задачи с ограничениями.



# Задачи с ограничениями

Задача ограничениями:

- множество переменных (сенсор, вершина);
- множества возможных значений для каждой переменной (частота, цвет);
- множество ограничений (пересекающиеся сенсоры, смежные вершины);
- необходимо назначить каждой переменной значение, чтобы ограничения были выполнены.

# Задачи с ограничениями

Задача ограничениями:

- множество переменных (сенсор, вершина);
- множества возможных значений для каждой переменной (частота, цвет);
- множество ограничений (пересекающиеся сенсоры, смежные вершины);
- необходимо назначить каждой переменной значение, чтобы ограничения были выполнены.

# Задачи с ограничениями

Задача ограничениями:

- множество переменных (сенсор, вершина);
- множества возможных значений для каждой переменной (частота, цвет);
- множество ограничений (пересекающиеся сенсоры, смежные вершины);
- необходимо назначить каждой переменной значение, чтобы ограничения были выполнены.

# Задачи с ограничениями

Задача ограничениями:

- множество переменных (сенсор, вершина);
- множества возможных значений для каждой переменной (частота, цвет);
- множество ограничений (пересекающиеся сенсоры, смежные вершины);
- необходимо назначить каждой переменной значение, чтобы ограничения были выполнены.

## Задача **оптимизации**:

- множество переменных (сенсор, вершина);
- множества возможных значений для каждой переменной (частота, цвет);
- множество ограничений (пересекающиеся сенсоры, смежные вершины);
- функция веса для каждого нарушенного ограничения;
- необходимо минимизировать сумму весов нарушенных ограничений;

## Задача оптимизации:

- множество переменных (сенсор, вершина);
- множества возможных значений для каждой переменной (частота, цвет);
- множество ограничений (пересекающиеся сенсоры, смежные вершины);
- функция веса для каждого нарушенного ограничения;
- необходимо минимизировать сумму весов нарушенных ограничений;

## Задача **оптимизации**:

- множество переменных (сенсор, вершина);
- множества возможных значений для каждой переменной (частота, цвет);
- множество ограничений (пересекающиеся сенсоры, смежные вершины);
- функция веса для каждого нарушенного ограничения;
- **необходимо минимизировать сумму весов нарушенных ограничений;**



Задача **оптимизации**:

- множество переменных (сенсор, вершина);
- множества возможных значений для каждой переменной (частота, цвет);
- множество ограничений (пересекающиеся сенсоры, смежные вершины);
- функция веса для каждого нарушенного ограничения;
- необходимо минимизировать сумму весов нарушенных ограничений;

**Задачи с ограничениями в общем случае NP-полные!**

# Распределенные задачи с ограничениями

В распределенной задаче оптимизации/с ограничениями переменные распределены между агентами.

Обычно каждый агент отвечает за одну переменную.

Задачи ИИ часто формулируются в виде задач с ограничениями.

Подходы к решению:

- адаптация классических решений (перебор, нахождение решения во всём пространстве поиска);
- адаптация алгоритмов локального поиска (нахождение оптимального решения в локальной области пространства поиска);
- кооперативные подходы (в основном заимствованные из природы/социологии).

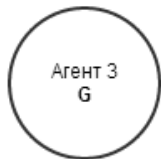
# Асинхронный перебор: основные концепции



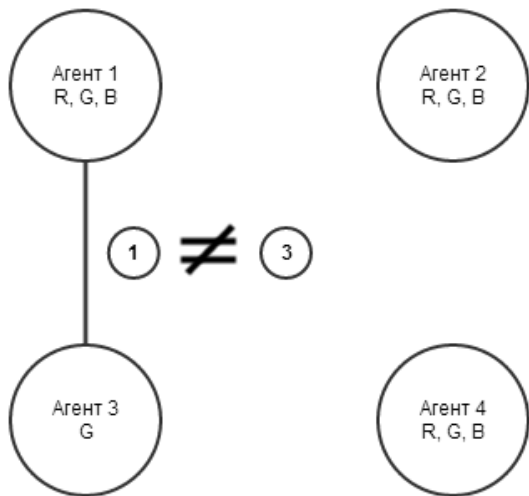
# Асинхронный перебор: основные концепции



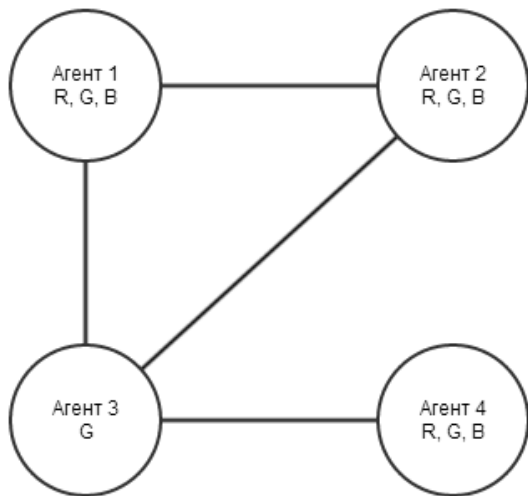
# Асинхронный перебор: основные концепции



# Асинхронный перебор: основные концепции

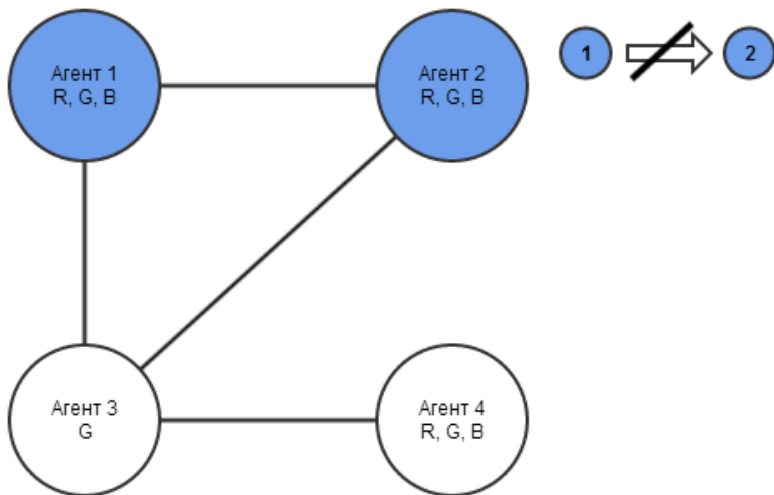


# Асинхронный перебор: основные концепции

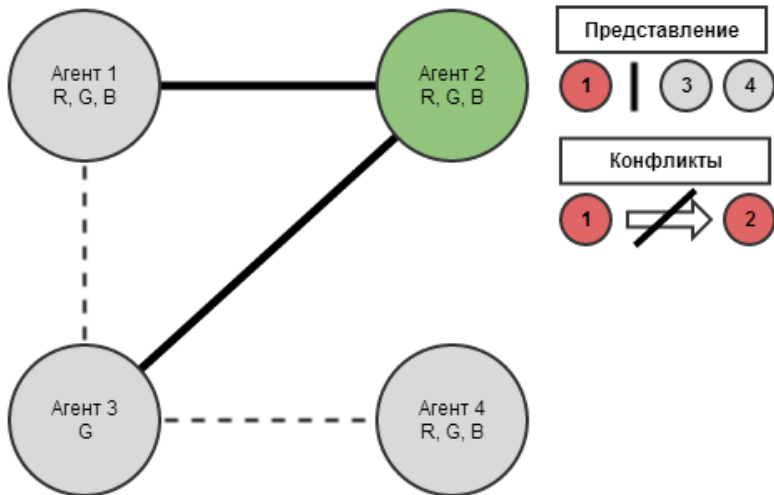




# Асинхронный перебор: основные концепции



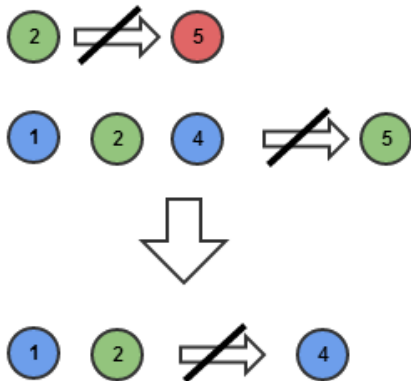
# Асинхронный перебор: основные концепции



# Асинхронный перебор: разрешение конфликтов



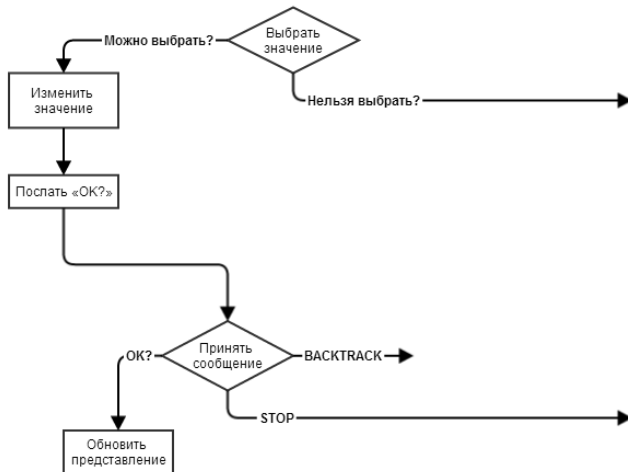
# Асинхронный перебор: разрешение конфликтов



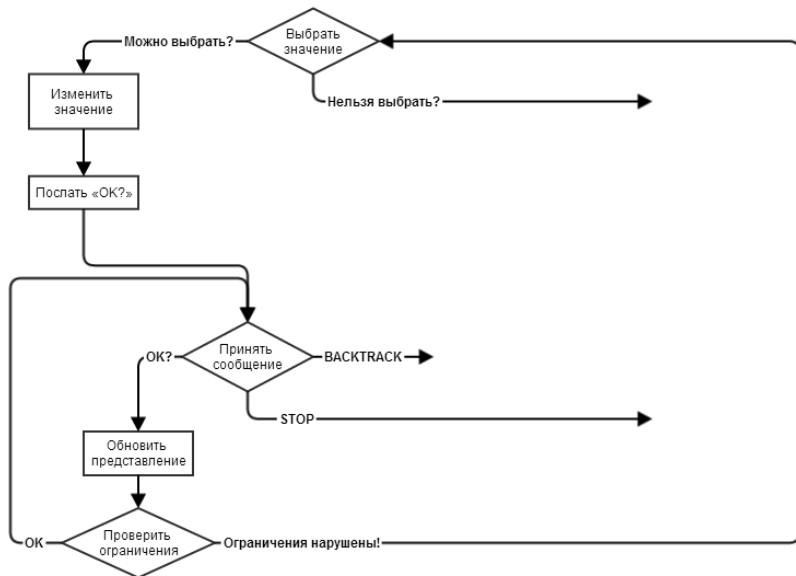
# Асинхронный перебор: алгоритм



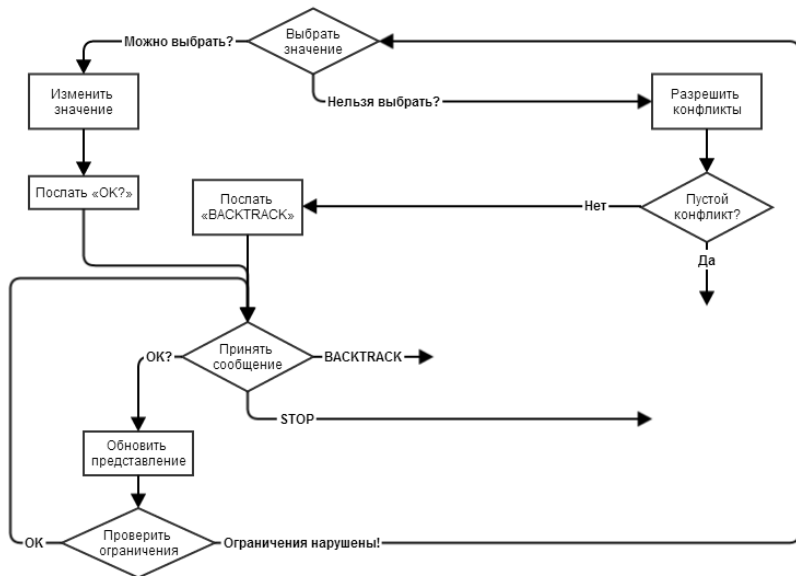
# Асинхронный перебор: алгоритм



# Асинхронный перебор: алгоритм

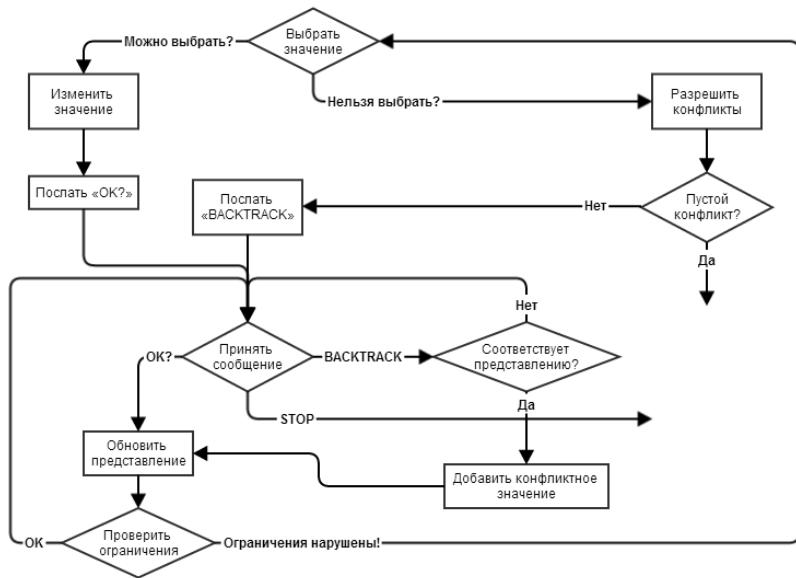


# Асинхронный перебор: алгоритм

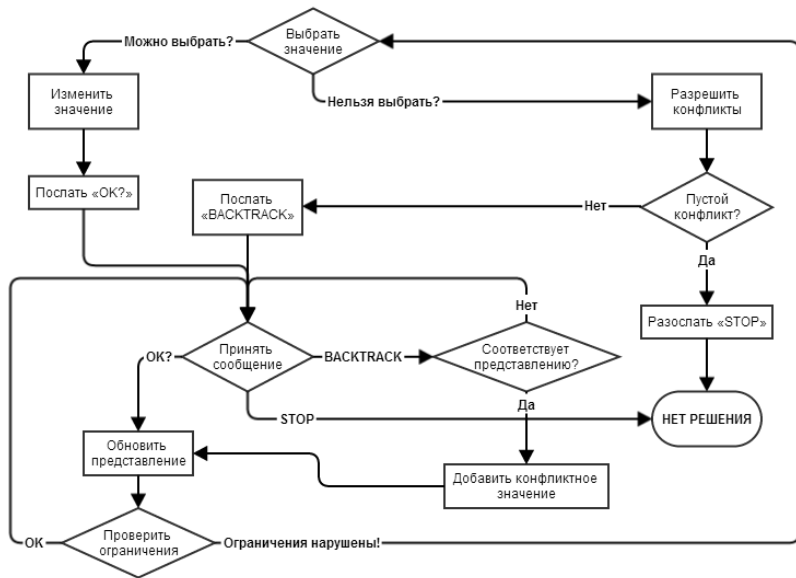




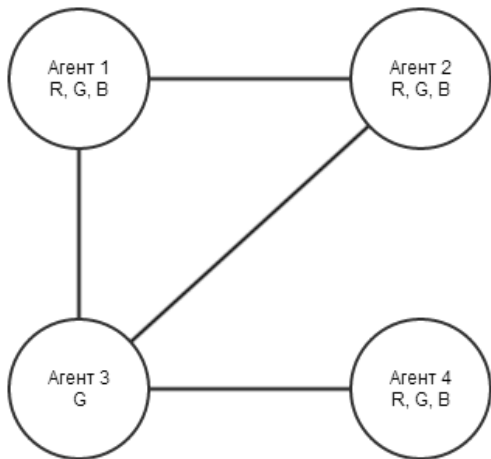
# Асинхронный перебор: алгоритм



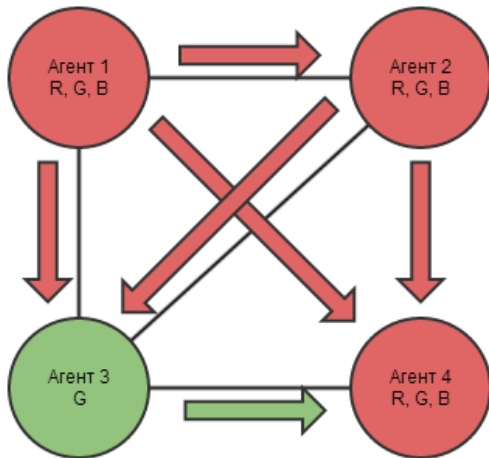
# Асинхронный перебор: алгоритм



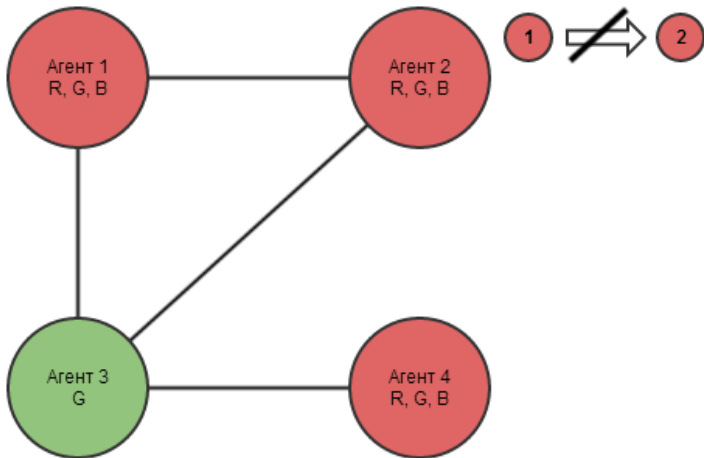
# Асинхронный перебор: раскраска графа



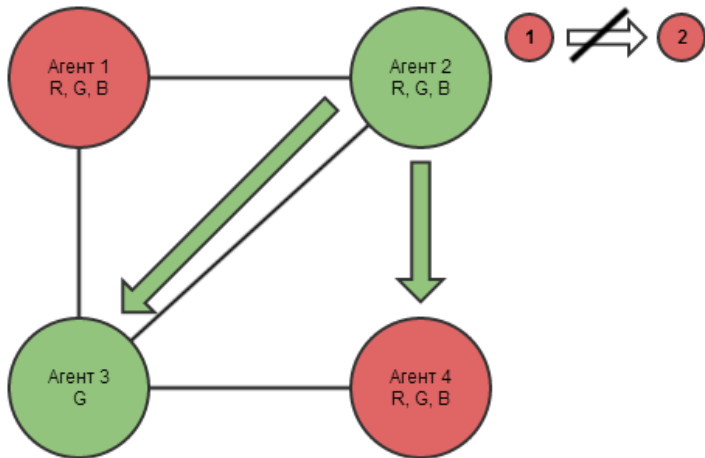
# Асинхронный перебор: раскраска графа



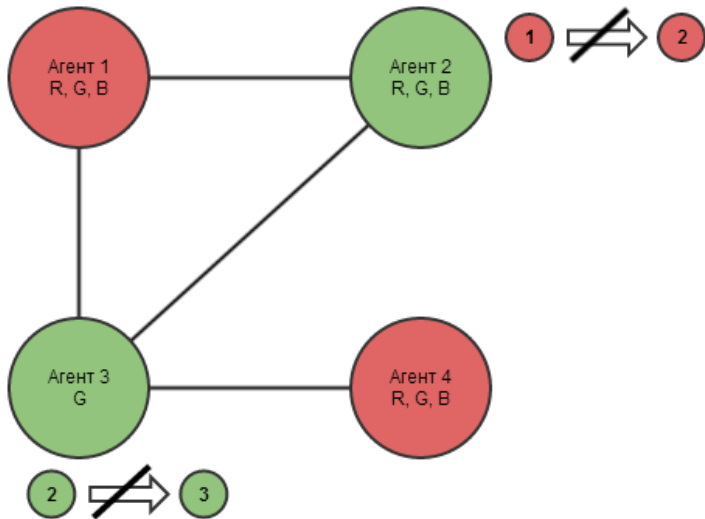
# Асинхронный перебор: раскраска графа



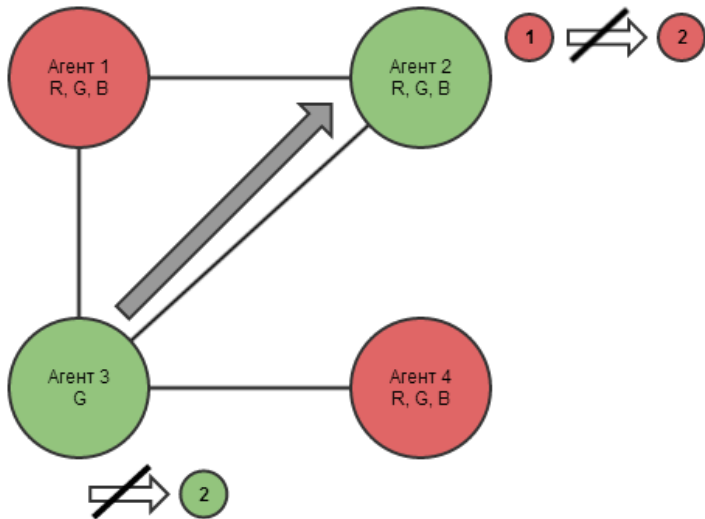
# Асинхронный перебор: раскраска графа



# Асинхронный перебор: раскраска графа

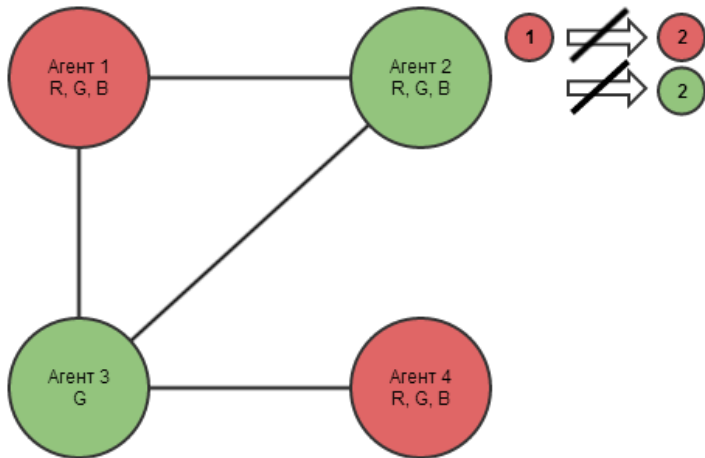


# Асинхронный перебор: раскраска графа

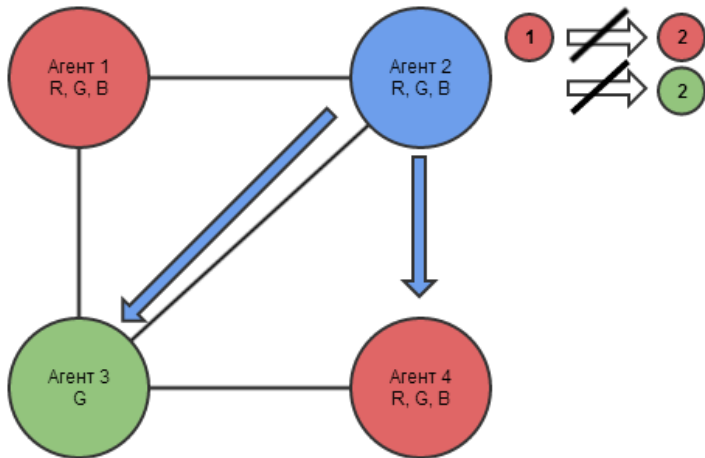




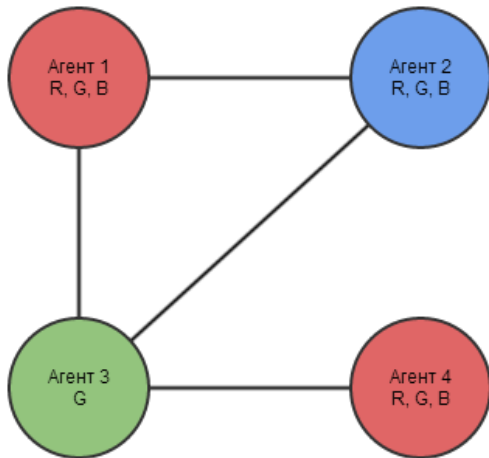
# Асинхронный перебор: раскраска графа



# Асинхронный перебор: раскраска графа



# Асинхронный перебор: раскраска графа



# Асинхронный перебор: итоги

Каждый агент отвечает за одну переменную и предлагает другим агентам принять значение, которое он выбрал.

- асинхронный перебор [BMBM05] — полный алгоритм;
- глобальное упорядочивание агентов;
- нет процедуры останова (но она легко добавляется);
- хорошо масштабируется.

Расширения:

- изменение порядка при конфликтах (AWCS [Yok95]);
- решение задачи оптимизации (ADOPT [Mod03], APO [ML06]);
- динамическое добавление агентов (DynAPO).

## Другой подход: распределенный локальный поиск

Алгоритмы распределенного локального поиска исследуют возможные изменения состояния:

- всегда стремятся улучшить состояние (уменьшить количество конфликтов);
- естественным образом поддерживают динамику (добавление ограничений, агентов);
- эффективны по времени;
- не полны и требуют настройки параметров.

Известные алгоритмы:

- Tabu search [GL97];
- Simulated annealing [Rad12];
- Iterative Breakout method [HY05].

# Другой подход: кооперативные алгоритмы

Популяция — это набор индивидуальных агентов.

- агент знает целиком исходную задачу (и может решить сам);
- агенты координируются для нахождения решения.

Известные алгоритмы:

- эволюционные и генетические алгоритмы [DCB<sup>+</sup>07, Han03];
- Particle Swarm Optimization [PV02];
- Ant Colony Optimization [DS02].

- 1 Концепция агента
- 2 Приложения МАС
- 3 Автономность в МАС**
- 4 Разработка многоагентных систем
- 5 Список литературы

# Мотивирующий пример: устранение ЧС

Пример: авария с возгоранием в туннеле. Вовлечены несколько организаций:

- пожарная бригада;
- скорая помощь;
- полиция;
- администрация;
- участники происшествия:
  - ▶ возможные жертвы;
  - ▶ наблюдатели;
  - ▶ участники дорожного движения.

Все участники имеют общую цель: устранение ЧС. Обмен информацией важен для корректной координации и технологии играют существенную роль.

При обсуждении проблем координации используется понятие автономности.



Как и для агента, не существует общепризнанного определения автономности:

- автономность — ключевая особенность агентов;
- в переводе с греческого означает «возможность самоуправления».

В многоагентных системах:

- агент должен проявлять социальную активность и учитывать других участников;
- агент должен быть ответственен за свои действия;
- баланс между этими требованиями приводит к появлению “уровней автономности” и контролируемой автономности.

“*Контролируемая автономность* — это способность динамически обрабатывать внешние воздействия с учётом внутренних мотиваций.”

— Bob van der Vecht [Vec09]

Контролируемая автономность означает способность агентов изменять уровень своей (или чужой) автономности:

- это одно из основных средств самоорганизации системы;
- делает прозрачным создание адаптивных систем;

# Уровни абстракции при принятии решений

Автономность рассматривается в контексте взаимодействия с другими агентами [FC01] (старшие агенты — на более высоком уровне абстракции, делегаты — на более низком):

- уровень автономности определяет, за какие решения отвечает делегат, а за какие — старший агент;
- уровни автономности:
  - ▶ исполнительный  
(каким образом выполнить задачу);
  - ▶ планирующий  
(планирование задач для достижения цели);
  - ▶ целевой  
(создание и изменение целей);
  - ▶ моральный  
(когда и как менять уровень автономности) [Ver00].

Агент может принимать или отказываться от порученных заданий [FC01]:

- чем менее специфицированную задачу может принять агент, тем выше его уровень автономности;
- изменение автономности — изменение зависимостей от других агентов;
- причины для смены уровня автономности:
  - ▶ агент не уверен, что может справиться с задачей;
  - ▶ агент предвидит проблемы, которые могут возникнуть;
  - ▶ агент считает, что может лучше справляться с большей автономностью;
  - ▶ агент хочет “удивить” старшего сверхурочной работой.

Автономность определяется свободой выбора и исполнения действий [BFJ<sup>+</sup>04].

- два аспекта: **возможность выполнить действие и разрешение на выбор** действия;
- действия различаются на
  - ▶ потенциальные (возможные в данной среде);
  - ▶ практически возможные (в текущем состоянии мира);
  - ▶ возможные (для данного агента)
  - ▶ если возможные действия агента совпадают с “практически возможными”, он считается полностью автономным (по первому аспекту).

Автономность измеряется вкладом в групповое решение [MB06]:

- автономность агента определяется для каждого задания;
- в отношении задания у некоторых агентов может быть приоритет в плане принятия решения;
- уровень автономности определяется степенью независимости решения агента от решения любого другого агента.

Пример: агенты должны решить, в каком порядке они будут эксклюзивно пользоваться общим ресурсом.

Агент может передавать управление процессом принятия решения [SST04] другому агенту или человеку.

- агенты располагают набором стратегий для передачи управления;
- агенты могут менять ограничения (например, требовать дополнительное время на принятие решения);
- временные рамки определяют моменты передачи управления.

Важной проблемой является осознание агентом своей автономности:

- осведомлённость о собственной автономности определяет основу ответственности;
- предполагается, что агент действует рационально и осознает свой выбор с учётом моральных законов;
- агент должен осознавать последствия принимаемых решений.

Агент, осознающий свою автономность не просто переключается между “режимами” работы, но оценивает последствия того или иного решения.



## Проблемы:

- пилоты коммерческих рейсов вынуждены полагаться на авиадиспетчеров;
- чтобы авиадиспетчеры могли справляться со своей задачей, в воздушном пространстве “вычерчиваются” специальные магистрали;
- самолёты регулярно стоят в очередях на такую магистраль;
- самолёты на магистрали должны поддерживать установленную скорость, чтобы избежать задержек;
- со временем растёт потребность в воздушном транспорте и, соответственно, увеличиваются задержки.

## Совместное управление воздушным траффиком [WSJ08]:

- моделирование воздушного траффика;
- агенты — диспетчерские центры (центральный, региональные и каждой авиалинии);
- цель — максимально эффективно использовать магистрали;
- уровни автономности напрямую связаны с иерархией диспетчерских;
- в рамках одного уровня иерархии (одной компании) приоритеты могут меняться;
- контроль автономности производится людьми.

# Содержание

- 1 Концепция агента
- 2 Приложения МАС
- 3 Автономность в МАС
- 4 Разработка многоагентных систем**
- 5 Список литературы

## Основные архитектуры агентов [WJ95]:

- *рассуждающие агенты* используют символичные вычисления и логический вывод для решения задач. Основные проблемы:
  - ▶ представление задачи в логической системе;
  - ▶ эффективная реализация процесса вывода.
- *реактивные агенты* используют набор поведений для эффективной работы в реальном мире. Полагается на эвристические алгоритмы.
- *гибридные архитектуры* объединяют преимущества обоих подходов.

При создании структуры МАС используются:

- *организационные шаблоны* [НСУ99]  
(формирование известных социальных структур);
- *фрактальные системы* [FSS03]  
(агенты объединяются в группы, которые действуют как единый агент);
- *поведенческие паразиты* [SRR07]  
(среда воздействует на внутреннее состояние агента, имитируя симбиоз).

# Средства разработки МАС

Существует множество средств разработки МАС. Из них стоит выделить ZAPL, Jason, Jack, IMPACT, JADE, Jadex, DEFACTO, ARTIMIS.

Большинство общих фреймворков нацелено на использование ментальных характеристик агента. Они предоставляют возможность задания целей агента при помощи специальных языков вроде AgentSpeak [Rao96].

Некоторые среды позволяют упростить программирование процесса принятия группового решения.

К сожалению, ни одна среда не предоставляет библиотеки распределенных алгоритмов, структурных или архитектурных шаблонов.

Спасибо за внимание!

- 1 Концепция агента
- 2 Приложения МАС
- 3 Автономность в МАС
- 4 Разработка многоагентных систем
- 5 Список литературы



# Список использованной литературы I



Jeffrey M. Bradshaw, Paul J. Feltovich, Hyuckchul Jung, Shrinivas Kulkarni, William Taysom, and Andrzej Uszok.

Dimensions of adjustable autonomy and mixed-initiative interaction.  
*In M. Klusch, G. Weiss, and M. Rovatsos (Ed.), Computational  
Autonomy*, 2969:17–39, 2004.



Christian Bessière, Arnold Maestre, Ismel Brito, and Pedro Meseguer.

Asynchronous backtracking without adding links: a new member in the  
abt family.

*Artif. Intell.*, 161(1-2):7–24, January 2005.



Gerry Dozier, Hurley Cunningham, Winard Britt, Yu Wang, Cheryl  
Seals, and Funing Zhang.

Distributed constraint satisfaction, restricted recombination, and  
genetic protocols.

*Appl. Soft Comput.*, 7(3):1005–1011, June 2007.

## Список использованной литературы II



Marco Dorigo and Thomas Stützle.

The ant colony optimization metaheuristic: Algorithms, applications, and advances.

In *Handbook of Metaheuristics*, pages 251–285. Kluwer Academic Publishers, 2002.



R. Falcone and C. Castelfranchi.

The human in the loop of a delegated agent: the theory of adjustable social autonomy.

*Trans. Sys. Man Cyber. Part A*, 31(5):406–418, September 2001.






Klaus Fischer, Michael Schillo, and Jörg Siekmann.




Holonic multiagent systems: A foundation for the organisation of multiagent systems.

In *Proceedings of the First International Conference on Applications of Holonic and Multiagent Systems (HoloMAS'03)*, 2003.

## Список использованной литературы III

-  Shaw Green, Leon Hurst, Brenda Nangle, Pádraig Cunningham, Dr. Pádraig Cunningham, Richard Evans, and Fergal Somers.  
Software agents: A review, 1997.
-  Fred Glover and Manuel Laguna.  
*Tabu Search*.  
Kluwer Academic Publishers, Norwell, MA, USA, 1997.
-  Hisashi Handa.  
Hybridization of estimation of distribution algorithms with a repair method for solving constraint satisfaction problems.  
*In Proceedings of the 2003 international conference on Genetic and evolutionary computation: Part I, GECCO'03*, pages 991–1002, Berlin, Heidelberg, 2003. Springer-Verlag.

## Список использованной литературы IV

-  Sandra C. Hayden, Christina Carrick, and Qiang Yang.  
Architectural design patterns for multiagent coordination.  
*In IN PROC. OF THE 3RD INT. CONF. ON AUTONOMOUS AGENTS, AGENTS'99*, 1999.
-  Katsutoshi Hirayama and Makoto Yokoo.  
The distributed breakout algorithms.  
*Artif. Intell.*, 161(1-2):89–115, January 2005.
-  Cheryl Martin and K. Suzanne Barber.  
Adaptive decision-making frameworks for dynamic multi-agent organizational change.  
*Autonomous Agents and Multi-Agent Systems*, 13(3):391–428, November 2006.

## Список использованной литературы V



Roger Mailler and Victor R. Lesser.

Asynchronous partial overlay: a new algorithm for solving distributed constraint satisfaction problems.

*J. Artif. Int. Res.*, 25(1):529–576, April 2006.



Pragnesh Jay Modi.

*Distributed constraint optimization for multiagent systems.*

PhD thesis, Los Angeles, CA, USA, 2003.

AAI3133310.



K. E. Parsopoulos and M. N. Vrahatis.

Particle swarm optimization method in multiobjective problems.

In *Proceedings of the 2002 ACM symposium on Applied computing*, SAC '02, pages 603–607, New York, NY, USA, 2002. ACM.

# Список использованной литературы VI



Atanas Radenski.

Distributed simulated annealing with mapreduce.

In *Proceedings of the 2012t European conference on Applications of Evolutionary Computation, EvoApplications'12*, pages 466–476, Berlin, Heidelberg, 2012. Springer-Verlag.



Anand S. Rao.

Agentspeak(I): Bdi agents speak out in a logical computable language, 1996.



Rónán Mac Ruairí and Mark T. Keane.




An energy-efficient, multi-agent sensor network for detecting diffuse events.

In *Proceedings of the 20th international joint conference on Artificial intelligence, IJCAI'07*, pages 1390–1395, San Francisco, CA, USA, 2007. Morgan Kaufmann Publishers Inc.

## Список использованной литературы VII

-  Amit Shabtay, Zinovi Rabinovich, and Jeffrey S. Rosenschein.  
Behaviosites: a novel paradigm for affecting distributed behavior-from  
a healthy society to a wealthy society.  
*In Proceedings of the 4th international conference on Engineering  
self-organising systems, ESOA'06, pages 82–98, Berlin, Heidelberg,  
2007. Springer-Verlag.*
-  Paul Scerri, Katia Sycara, and Milind Tambe.  
Adjustable autonomy in the context of coordination.  
*In In AIAA 3rd "Unmanned Unlimited" Technical Conference,  
Workshop and Exhibit, 2004. Invited Paper, 2004.*
-  Bob van der Vecht.  
*Adjustable Autonomy: Controlling Influences on Decision Making.*  
PhD thesis, Utrecht University, June 2009.

## Список использованной литературы VIII

-  Henricus J.E. Verhagen.  
Norm autonomous agents, 2000.
-  Michael Wooldridge and Nicholas R. Jennings.  
Intelligent agents: Theory and practice.  
*Knowledge Engineering Review*, 10:115–152, 1995.
-  Shawn R. Wolfe, Maarten Sierhuis, and Peter A. Jarvis.  
To bdi, or not to bdi: design choices in an agent-based traffic flow management simulation.  
*In Proceedings of the 2008 Spring simulation multiconference, SpringSim '08*, pages 63–70, San Diego, CA, USA, 2008. Society for Computer Simulation International.





Makoto Yokoo.

Asynchronous weak-commitment search for solving distributed constraint satisfaction problems.

*In Proceedings of the First International Conference on Principles and Practice of Constraint Programming, CP '95, pages 88–102, London, UK, UK, 1995. Springer-Verlag.*