



»Лекционен курс

»Интелигентни системи



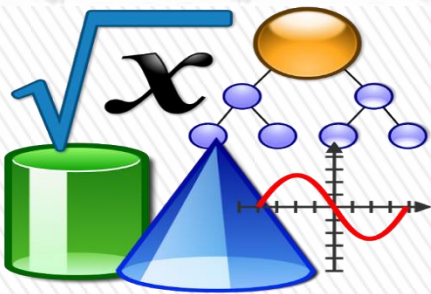
Решаване проблеми
чрез търсене



Решаване на проблеми



Каква е разликата?



АРП

- » Агентите трябва да намерят последователност от действия за постигане на целите си
- » Агенти с цели (за разлика от рефлексивните агенти)
 - > Имат предвид бъдещите действия и доколко задоволителни са резултатите
- » „Агент за решаване на проблеми“ (АРП)
 - > Един вариант на агент с цели
 - > Използват атомарни представяния, т.е. състоянията на света се разглеждат като едно цяло
- » „Планиращи агенти“ (ПА)
 - > Агенти с цели, които използват разширени факторизирани или структурирани представяния на състоянията



Формулиране на цел

- » Първата стъпка за решаване на проблема
- » Целите помагат за организация на поведението на агентите
- » Цел
 - > Множество от състояния
- » АРП трябва да разбере как да действа сега и в бъдеще за да постигне едно целево състояние
 - > За да може да направи това трябва да реши кои действия и състояние ще взема под внимание при опитите за достигане на целта



Формулиране на проблема

» Формулиране на проблема

- > Процес за определяне действията и състоянията, които трябва да се имат предвид за дадена цел

» АРП не знае кое от възможните действия е най-доброто

- > Понеже не знае достатъчно за състоянията, които резултат от изпълнението на отделните действия

» Ако не притежава допълнителна информация (т.е. средата е непозната), АРП няма друг изход освен случайно да изпробва едно действие

» Ако притежава необходимите средства АРП може първо да изследва възможните опции, преди да пристъпи към реално изпълнение на действията



Допускания за средата

» Наблюдаема:

- > Агентът винаги знае актуалното състояние

» Дискретна:

- > Във всяко състояние има краен брой действия

» Позната:

- > Агентът знае кои състояния се достигат при извършване на действията

» Детерминирана:

- > Всяко действие поражда един и същ резултат

» При тези предпоставки решението на един проблем е фиксирана последователност от действия



Търсене

» Търсене

- > Намиране на последователност от действия, която достига целта

» Търсещ алгоритъм

- > Вход: проблем
- > Изход: решение, под формата на последователност от действия

» Фаза на изпълнение

- > След като е намерено едно решение могат да се изпълняват действията, които то предписва



АРП

function **Simple-Solving-Agent** (percept) returns едно действие
persistent: seq, последователност от действия, в началото празна
state, описание на актуалното състояние на света
goal, цел, в началото null
problem, една формулировка на проблем

state \leftarrow Update-State(state, percept);

if seq = \emptyset then {

goal \leftarrow Formulate-Goal(state);

Формулира цел

problem \leftarrow Formulate-Problem(state, goal);

Формулира проблем

seq \leftarrow Search(problem);

if seq = failure then return едно нулево действие

}

action \leftarrow First(seq);

Търси последователност от действия за решаване на проблема

seq \leftarrow Rest(seq);

return action

Отворен цикъл (open loop)

- » По време на изпълнение на последователността от действия, предписани в решението, един агент за решаване на проблеми (АРП) игнорира възприятията
 - > Приема се, че предварително знае техния ефект
 - > Т.е. един агент, който изпълнява плана със „затворени очи“ трябва да е много сигурен какво се случва
- » Такива системи се наричат с „отворен цикъл“
 - > Понеже игнорирането на възприятията прекъсва цикъла между агента и околната среда



Формално представяне на проблем

» Един проблем формално се дефинира посредством 5 компонента:

- > Начално състояние (НС)
- > Описание на възможните действия (ВД)
- > Модел на преходите (МП)
- > Целеви тест (ЦТ)
- > Функция на разходите за път (ФРП)



Начално състояние

» Начално състояние (НС)

- > Състояние, в което АРП започва търсенето
- > Пример: In(Пловдив)



Възможни действия

» Описание на възможните действия (ВД)

- > Операции, които може да извърши един АРП
- > Функция **Actions(s)**
 - + Връща множество на действията, които могат да бъдат извършени в състояние **s**
 - + Пример: $\text{Actions}(\text{In}(\text{Пловдив})) = \{\text{Go}(\text{СтараЗагора}), \text{Go}(\text{Асеновград}), \text{Go}(\text{Карлово})\}$



Модел преходи

» Модел на преходите (МП)

- > Описание на това, което предизвиква всяко действие
- > Специфицира се посредством функцията **Result(s, a)**
 - + Връща състоянието, резултиращо от извършване на действието **a** в състояние **s**
- > Наследник – състояние, достигнато от едно дадено състояние (предшественик) посредством прилагане на едно отделно действие
- > Пример: $\text{Result}(\text{In}(\text{Пловдив}), \text{Go}(\text{Карлово})) = \text{In}(\text{Карлово})$



Целеви тест

» Целеви тест (ЦТ)

- > Решава дали едно актуално състояние е целево
- > Може да се зададе по два начина:
 - + Явно – като множество от целеви състояния
 - Проверява дали актуалното е елемент от това множество
 - Пример: {In(Бургас), In(Варна)}
 - + Неявно – като абстрактно свойство
 - Шах: „Шах-мат“ – противниковия цар няма полезен ход



Разходи

- » Функция на разходите за път (ФРП)
 - > Присвоява на всеки път числена стойност на разходите
 - > АРП избира такава функция, която отразява неговата представа за необходимите разходи
 - > Например: време, количество гориво, ...
- » Разходите за едно действие (стъпка) a от състояние s към състояние s' представяме посредством $c(s, a, s')$



Пространство на състояния

- » Пространство на състоянията (ПС)
 - > Множество на всички състояния, достижими от НС посредством една последователност от действия
- » Дефинира се от НС, ВД и МП
- » Обикновено се визуализира като насочена мрежа или граф, където:
 - > Възли – представят състояния
 - > Ребра – представят действия
 - > Път – последователност от състояния, свързани с последователност от действия



Решение на проблем

» Решение:

- > Последователност от действия, водеща от начално състояние към целево състояние

» Качество на решение:

- > Измерва се на основата на функцията за разходи на път

» Оптимално решение:

- > Решението с най-малките разходи от всички решения



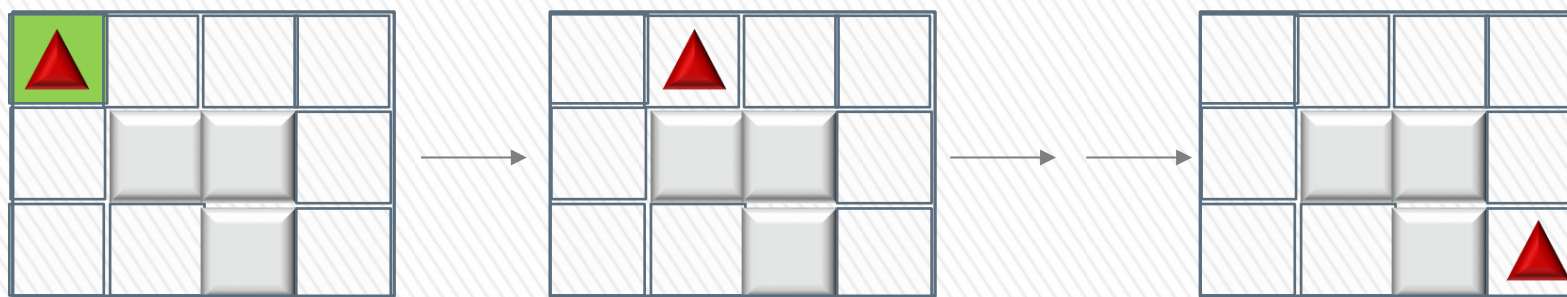
Формулиране на проблеми

- » Абстракция
- » Премахване детайли от представянето за:
 - > Състоянията
 - > Действията
- » Добра абстракция
 - > Премахване по възможност най-много детайли при запазване валидността на проблема



Пример

Оператори: преместване в съседен квадрат



Цел: долен десен квадрат

Допускане: детерминирана среда (знаем въздействието на операторите в средата)

Пространство на състояния (ПС)

- » Състояние: $s \in \text{States}$, States – множество на състояния
- » Оператори: $a \in \text{Actions}(s)$
- » Наследник: $\text{Succ}(s,a) \in \text{States}$
- » Разходи: $c(s,a,s') \in \mathbf{R}$
- » Начално състояние: $s_{\text{start}} \in \text{States}$
- » Целеви тест: $\text{IsGoal}(s)$

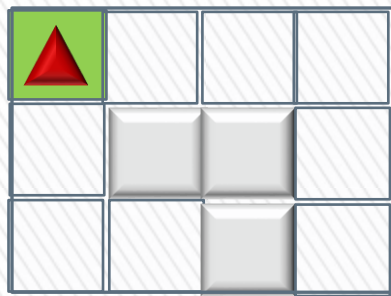


Тест: модел за търсене на път

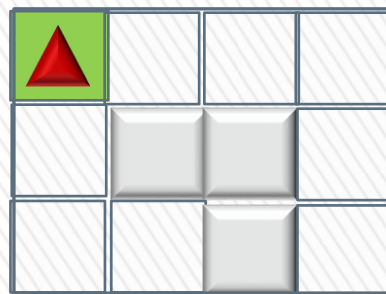


Какви стойности?

$s_{start} =$



$Cost($



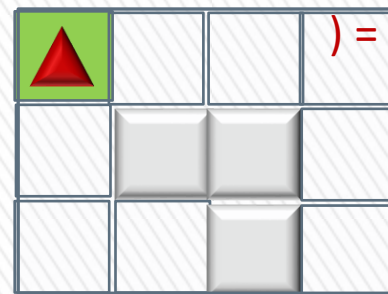
$, \text{изток}) =$

1

$Actions(s_{start}) =$

„ИЗТОК“

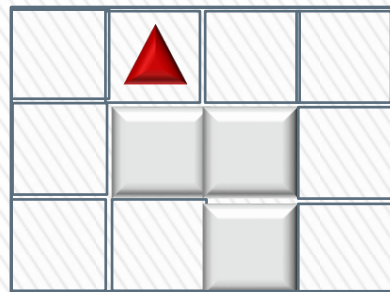
$IsGoal($



$) =$

false

$Succ(s_{start}, a) =$



Оптимизационен проблем

- » Да се намери един път (последователност от действия)
 - > $p = (a_1, \dots, a_n)$ с минимални разходи
 - > Разходи за един път $\text{PathCost}(p) = \sum c(s_{i-1}, a_i, s_i)$
 - > Ако p достигне целта:
 - + $[s_0 = s_{\text{start}}, s_i = \text{Succ}(s_{i-1}, a_i), \text{IsGoal}(s_n) = \text{true}]$
 - > В противен случай
 - + Безкрайни разходи



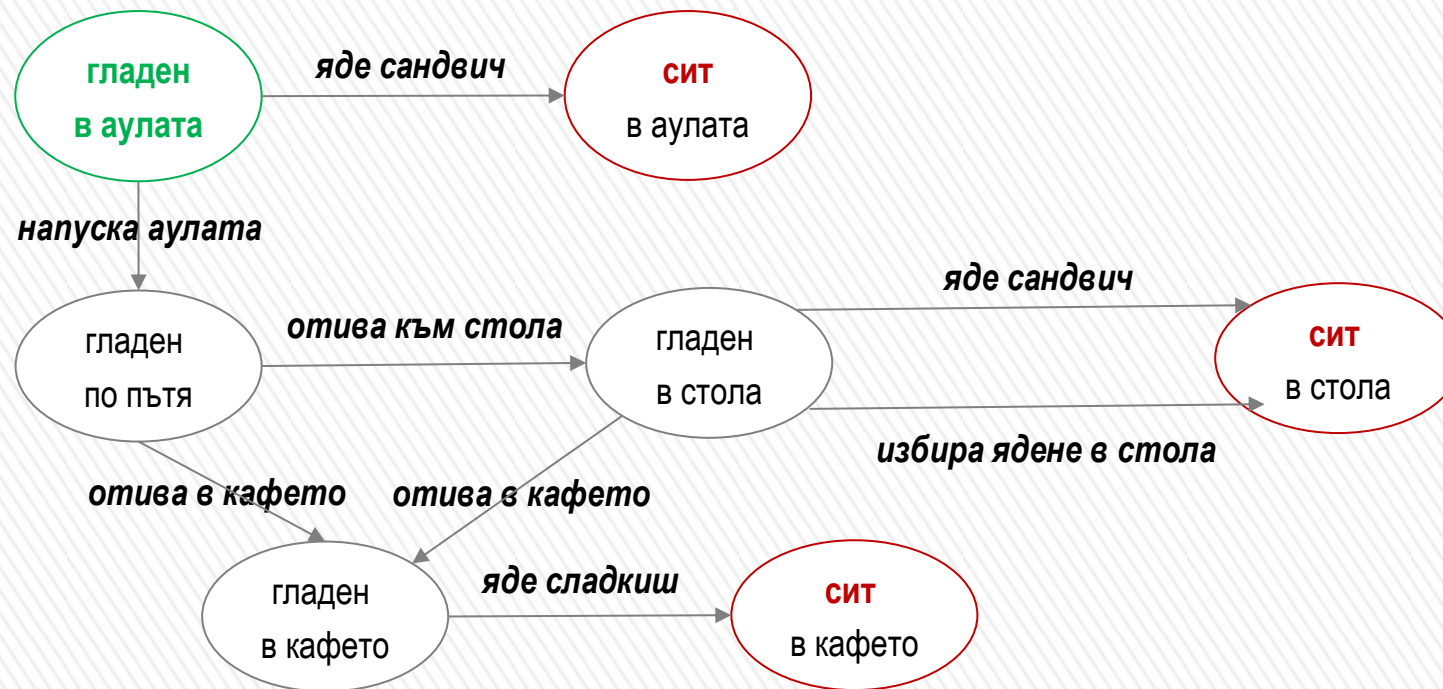
Представяне като граф

- » В много случаи е удобно ПС да се представя като граф, където:
 - > Всеки възел е състояние $s \in \text{States}$
 - > Всяко (насочено) ребро е двойката $(s, \text{Succ}(s,a))$ с разходи $c(s,a,s')$ за действие $a \in \text{Actions}(s)$
 - > Целеви възли: подмножество на възлите, което удовлетворява IsGoal
 - > Оптимизационен проблем: да се намери път от началния възел към един целеви възел с минимални разходи



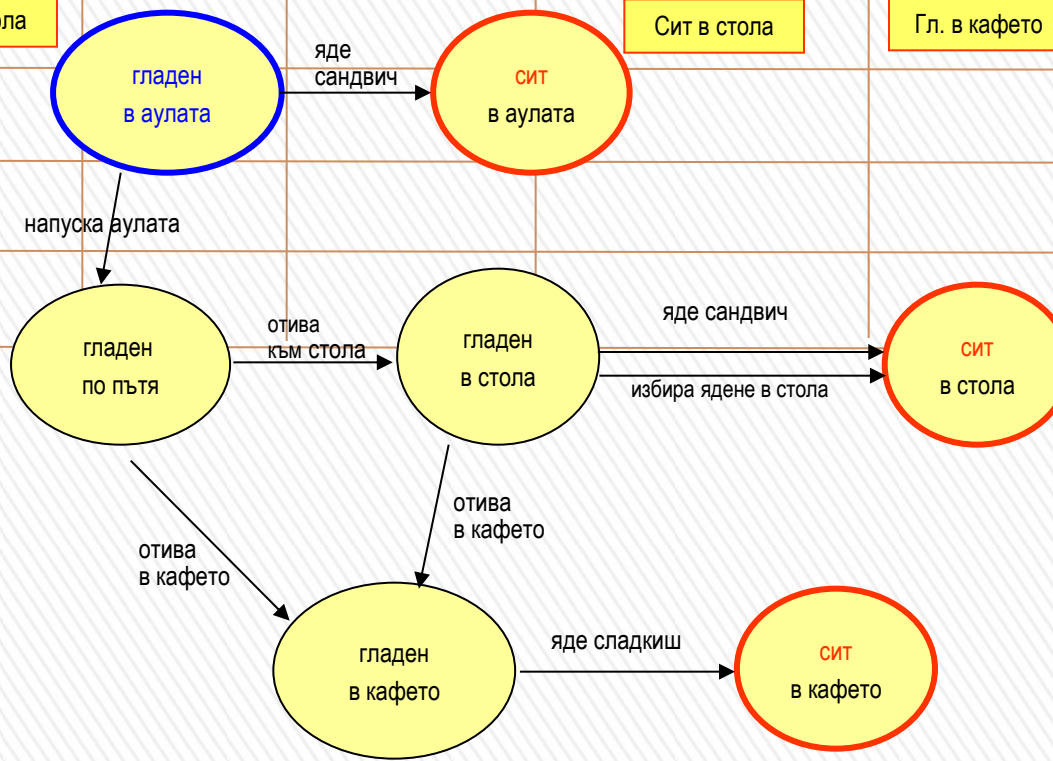
Пример: моделиране като граф

- » Начално състояние: { гладен, в аулата }
- » Условие за целеви състояния: { СИТ }



Като матрица на преходи

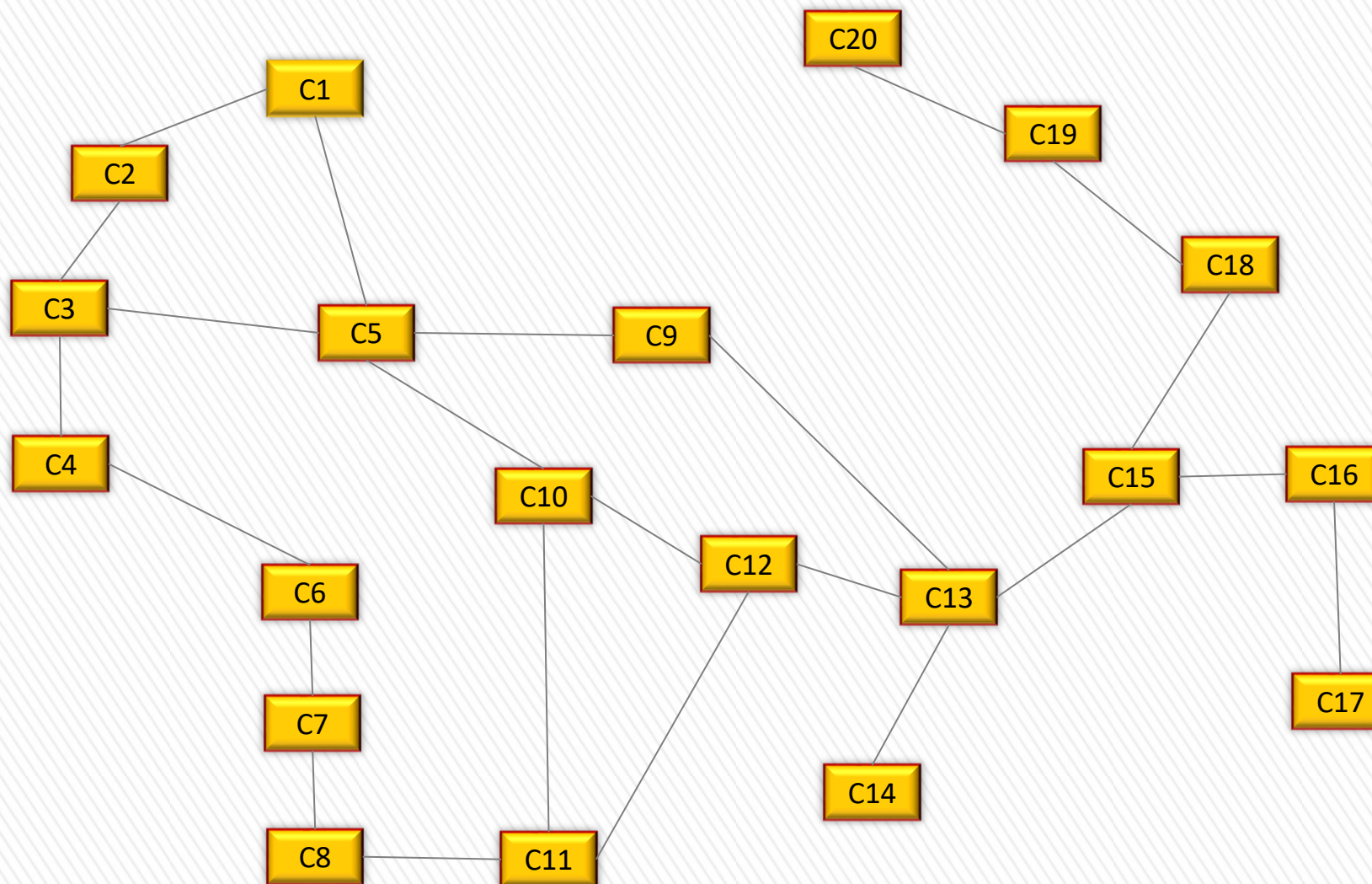
	Яде сандвич	Напуска аулата	Отива към стола	Избира ядене в стола	Отива в кафето	Яде сладкиш
Гладен в аулата	Сит в аулата	Гл. по пътя				
Гладен по пътя			Гл. в стола		Гл. в кафето	
Сит в аулата						
Гладен в стола	Сит в стола			Сит в стола	Гл. в кафето	
Гладен в кафето						Сит в кафето
Сит в аулата						
Сит в кафето						



- Граф
- Транзактивна система
- Автомат
- Акцептор



Пример: пътна карта



Комплексност (брой на състояния/възли)

- 8-ка пъзел: $9!$ състояния
от тях $9!/2 = 181.440$ достъпни
- 15-ка пъзел: $16!$ състояния
от тях $16!/2$ достъпни
- Унгарски куб: $12 \cdot 4,3 \cdot 10^{19}$ състояния
 $1/12$ от тях достъпни: $4,3 \cdot 10^{19}$
- Ханойски кули: 3^n състояния за n пула
решение след $(2^n) - 1$ хода
- Дама: около 10^{40} игри със средна продължителност
- Шах: около 10^{120} игри със средна продължителност
- Го: 3^{361} положения



Проблеми с комплексността


- » Недостатъчна памет за пространството на състоянията
- » Разходи за разпознаване на повторенията

Метод за решение: „Разширяване пространството на състоянията“:

Постъпково конструиране и изследване на състоянията

Конструиране – Тестване –
Забравяне





Благодаря за вниманието!