

Лекция 2: **Търсене на път до определена цел**

Характеристики на алгоритмите за търсене:

- пълнота
- оптималност
- сложност
 - по време \approx брой изследвани възли
 - по памет \approx максимален размер на фронта

Необходимост от избягване на повторения и зацикляне.

Параметри на дървото на търсене:

- дълбочина (височина) - d
- коефициент на разклонение (разклоненост) - b

Алгоритми за неинформирано ("сляпо") търсене

Обща характеристика: пълно изчерпване по твърда (фиксирана отнапред) стратегия. Прилагат се, когато липсва специфична информация за предметната област и оценяващата функция може само да провери дали дадено състояние е целево или не е.

Програмна реализация: чрез използване на работна памет (списък Open на т. нар. открити възли или списък от натрупани/изминати пътища, започващи от началния възел – нарича се още фронт на дървото на търсене).

1. Изчерпване (търсене) в дълбочина (depth-first search) - добавяне на новите възли (новия възел) в началото на списъка Open.

Търсенето е евтино (линейно), но не е нито пълно, нито оптимално (пълно е, когато графът на състоянията е краен).

Вариант 1: търсене в дълбочина до определено ниво (depth-bound search).

Вариант 2: итеративно търсене по нива (iterative deepening).

2. Изчерпване (търсене) в широчина (breadth-first search) - добавяне на новите възли в края на списъка Open.

Търсенето е пълно и оптимално, но е скъпо (експоненциално).

Вариант: търсене с равномерна цена на пътя (uniform cost search) - сортиране на списъка Open според цената на изминатия път.

Алгоритми за информирано (евристично) търсене

Обща характеристика: реализират пълно изчерпване по гъвкава стратегия или търсене с отсичане на част от графа на състоянията. Приложими са при наличие на специфична информация за предметната област, позволяваща да се конструира оценяваща функция (евристика), която връща небулева оценка (числова оценка в предварително определен интервал). Тази оценка може да служи например за мярка на

близостта на оценяваното състояние до целта или на необходимия ресурс за достигане от оценяваното състояние до целта.

Програмна реализация: чрез използване на работна памет (списък Open на т. нар. открити възли или списък от натрупани/изминати пътища, започващи от началния възел – нарича се още фронт на дървото на търсене).

1. Метод на най-доброто спускане (търсене на най-добър път, best-first search) - сортиране на списъка Open в съответствие с евристичната функция (например $h(node) = \text{<оценка на цената на пътя от } node \text{ до целта>}$).
Методът е ефективен, но не е нито пълен, нито оптимален.
2. Търсене с ограничена широчина (търсене в лъч, beam search) - ограничаване на списъка Open до първите n най-добри възела от него (в съответствие с евристиката). При $n=1$ се доближава до метода на най-бързото изкачване.
3. Метод на най-бързото изкачване (hill climbing) - списъкът Open се ограничава до най-добрия му елемент (в съответствие с евристиката), и то само ако той е по-добър от своя родител. Търсенето е еднопосочно, без възможност за възврат.

Възможни проблеми:

- *локален екстремум* – състоянието е по-добро от съседните (наследниците си), но не е най-доброто в цялото РС;
 - *плато* – съседните състояния (наследниците на текущото състояние) изглеждат също толкова добри, колкото и текущото;
 - *хребет* – никой от възможните оператори не води до по-добро състояние от текущото, макар че два или повече последователни оператори биха могли да доведат до такова състояние.
4. Търсене с минимизиране на общата цена на пътя (A^*) - комбинира търсене с равномерна цена на пътя с търсене на най-добър път. Списъкът Open се сортира в съответствие с функцията $f(node) = g(node) + h(node)$. Тук функцията g връща като резултат цената на изминатия път от началния възел до $node$, а евристичната функция h връща като резултат приблизителна стойност на цената на оставащата част от пътя от $node$ до целта.

Стратегията е пълна, ако разклоненията (наследниците) на всеки възел от РС са краен брой и цената на преходите е положителна, и оптимална, ако евристиката е *приемлива* (оптимистична), т.е. никога не надценява стойността (цената) h^* на оставащия път (ако $h^*(node) \geq h(node)$ за всеки възел $node$).