**10. Колко вида основни търсещи стратегии и алгоритми знаете? Обяснете с какво се различават:**

* **Глобално търсене** - разглежда се цялото пространство на състоянията. Ако се налага, може да се мине през всички състояния.
* **Локално търсене** - Разглежда се само локалното пространство, така че се гледат само състоянията, които са там. Ако решението е извън това пространство, няма как да се намери.
* **Неинформирани търсещи стратегии:** Неинформираните търсения използват само информацията, предоставена в самата дефиниция на проблема. Примери за неинформирани търсения са: DFS, BFS, UCS, DLS, IDS
* **Информирани търсещи стратегии**: Информираните стратегии за търсене имат информация за целевото състояние, което помага за по-ефикасно търсене. Тази информация се взема от евристична функция, която изчислява колко близо е сегашното състояние до целевото такова. Примери за информирано търсене са: Greedy Best-First Search, A\*, Beam Search, Hill Climbing

**6. Комбинация от кои алгоритми е „Итеративното търсене в дълбочина“. Изброите предимствата и недостатъците му (какво от кой алгоритъм взима).**

Пълен и оптимален, ако теглото на ребрата е еднакво. Комбинация от BFS и DFS като взима от предимствата и на двата. По-бърз е от BFS и хаби по-малко памет. Подходящ е за неинформирано търсене в огромни пространства, където дълбочината на решението не се знае.

**7. Каква е връзката между Uniform Cost Search и Dijkstra.**

Dijkstra е вариант на Uniform Cost Search, при който няма целево състояние и продължава докато не премахне всички върхове от приоритетната опашка (открива най-краткия път до всеки един връх).

**9. На какво е равна оценяващата функция на A\*: f(n) = ? Обяснете.**

f(n) = d(n) + h(n) -> Реалния път до текущия връх n плюс евристичната функция определяща цената на пътя до най-близкия целеви връх.

**10. Какво е евристична функция и какво е най-важното за нея?**

Да е равна на 0 в целевите състояния и изчисляването и да не е свързано с голям разход на време и памет.

**12. Какво представлява задачата за Удовлетворяване на ограниченията (Constraint Satisfaction Problem)?**

Математически проблем, определен като набор от обекти, чието състояние трябва да отговаря на редица от ограничения. Често проявяват висока сложност, изискваща комбинация от евристични и комбинаторни методи за търсене, за да бъде решен в рамките на разумен срок.

**13. Дайте примери за алгоритми, които се използват за решаването на проблеми от класа задачи CSP и съответно конкретни проблеми, които могат да се сведат до CSP.**

**Проблеми**: Map Coloring Problem; Sudoku; 8 queens.

Обикновено проблеми от тип Удовлетворяване на ограниченията се решават чрез използването на определена форма на търсене. Най-използваните варианти са Backtracking (Използва DFS), Constraint propagation и Local Search (Min-Conflicts).

**14. Какво е генетичен алгоритъм?**

Генетичен алгоритъм е търсеща евристика, която имитира процеса на естествения подбор. Тази евристика се използва рутинно за генериране на полезни решения за оптимизационни и търсещи проблеми. Генетичните алгоритми принадлежат към големия клас на еволюционните алгоритми, които генерират решения за оптимизационни проблеми използвайки техники, вдъхновени от естествената еволюция като наследяване,

мутация, селекция, кръстосване.

**15. Какво означава терминът еволюция в генетичните алгоритми?**

Еволюция е създаване на поколения по-добри индивиди посредством промяна чрез възпроизводство и избирателно оцеляване на част от наследниците.

**16. Изброите трите основни етапа при генетичните алгоритми и дайте определение за всеки един.**

**Селекция**:

В генерирането на състоянията от следващото поколение участват някои от най-добрите представители на текущото поколение (съгласно оценяващата функция), избрани на случаен принцип.

**Кръстосване**:

Чрез crossover се разглежда естествения дарвински модел на еволюция, при който характеристиките на двама или повече родители се кръстосва с цел да се произведе наследник с по-добър fitness от този на родителите си.

**Мутация**:

Извършване на случайни промени в случайно избрана малка част от новата популация с цел да се осигури възможност за достигане на всяка точка от пространството на състоянията и да се избегне опасността от попадане в локален екстремум.

**20. На какво се базира ефективността на алфа-бета отрязването?**

Реда на обхождане на наследниците.

**21. Сравнете алгоритмите greedy best first search и A\*. Прилики и разлики?**

Алгоритмите си приличат по това, че и двата са вариации на Best-First Search-a. Следователно и двата използват евристична функция върху всеки връх, за да определят кой ще бъде следващия посетен връх. И двата алгоритъма използват опашка, сортирана в намаляващ ред спрямо кой ще е най-скоро посетения връх. Различават се по това, че Greedy Best Search не е пълен и не е оптимален алгоритъм, докато А\* е пълен и оптимален алгоритъм. Също така Greedy Best Search алгоритъма взима върховете с най-ниска евристика(тези, които са най-близо до целта), докато А\* алгоритъма освен евристиката на даден връх, взема предвид и нужната цена, за да се стигне до него.

**22. Посочете възлите в дървото по-горе, чиито резултати не е нужно да бъдат изчислени. Обяснете защо.**

Възелът със стойност 2 и възелът със стойност -8 нямат нужда да бъдат оценявани, защото след -5 можем да сме сигурни, че няма как най-малкият елемент в другите поддървета да е едновременно по-малък от -5 и по-голям от -3.

Възелът със стойност 3 не е необходим, защото след -4 можем да сме сигурни, че няма как най-малкият елемент в другите поддървета да е едновременно по-малък от -4 и по-голям от -3.

**23. Сравнете алгоритмите търсене в лъч (beam search) и търсене в широчина (breadth-first)**

Breadth-first е неинформиран търсене, което е пълен, ако дълбочината d е крайна, но не е оптимален, освен ако цената на всяка стъпка не е константа. Beam search е информирано търсене, но не е пълен и оптимален. Едно предимство на Beam search е че използва по-малко памет от BFS, но не е пълен и е възможно да не разгледа някоя добра възможност заради лоша евристика на даден етап

**24. Сравнете следните формализми за представяне на знания: Предикатна логика и Семантични мрежи**

**Предикатна логика**:

По-експресивни са от съждителните логики. Синтаксисът им позволява константи, променливи, предикати, функции и квантори, докато семантиката им позволява истината на твърденията да се открие с дедуктивни правила. Програмирането е произлязло от логиката и има много математически трудове посветени на нея.

**Семантични мрежи:**

При семантичните мрежи лесно се следва йерархията и могат бързо да се правят асоциации. Също така лесно могат да се изменят. Минус обаче е че node-овете могат да имат много значения, трудно се програмира и се хващат грешки.

**25. Сравнете съждителните (пропозиционални) и предикатните логики от първи ред като формализми за представяне на знания. Посочете поне по едно предимство на всеки един.**

**Съждителни (пропозиционални) логики**:

Пропозиционалната логика е логика на сложните съждения. Тя разкрива тяхната структура и изяснява логическите операции, чрез които се образуват от прости съждения нови сложни съждения посредством логическите съюзи. Синтаксисът им включва конектори: и, или, не, предполага, еквивалентно и скоби, T (истина) и F (лъжа)

А при семантика истината на твърденията може да се открие с таблици на истината

**Предикатна логика**:

По-експресивни са от съждителните логики. Синтаксисът им позволява константи, променливи, предикати, функции и квантори, докато семантиката им позволява истината на твърденията да се открие с дедуктивни правила. Програмирането е произлязло от логиката и има много математически трудове посветени на нея.

Съждителните логики са по-прости са за програмиране, докато предикатните могат да се образуват по-сложни изречения.

**26. Сравнете генетичните алгоритми (Genetic algorithms) с алгоритъма за локално търсене в лъч (Local Beam search)**

И двата са локално търсещи и използват информирано търсене. Генетичните алгоритми запазват най-добрите си състояния и генерират нови състояния от тях, а beam search избира от текущия връх следващите N на брой най-добри. Също така генетичните алгоритми използват идеята за естествен подбор, докато beam search използва определена евристика.Генетичните алгоритми имат по-голям шанс да намерят решение, тъй като работят върху популация от решения и имат за задача да открият още по-добро такова. Докато локалното търсене в лъч има опасност да попадне в локални екстремуми, и да не намери решение или да му отнеме твърде много време преди да намери такова.

**16. Формулирайте теста на Тюринг за Изкуствения интелект:**

Теста на Тюринг е тест, искащ да покаже дали компютърът може да мисли и да се държи интелигентно (като човек). За теста се изисква един човек, който си взаимодейства дистанционно с човек и компютър.Човекът изпраща съобщение и получава отговор от компютъра или другия човек. Първия човек трябва да разпознае дали е получил отговора от компютъра или втория човек

**11. Какво е локално търсещ алгоритъм? Дайте пример за такъв алгоритъм.**

Локално търсещ алгоритъм е този, който търси в ограничено подпространство на цялото пространство на състоянията. По този начин не разглежда всички състояния и може да не намери решение. Примери: Hill Climbing, Beam Search..

**3. Как оценяваме дали даден алгоритъм се представя адекватно върху конкретен проблем (Метрики за оценяване)?**

Дали алгоритъмът е пълен и оптимален? Каква е сложността на алгоритъма по време/памет?

**7. Какво значи даден алгоритъм да е оптимален?**

Винаги да намира най-доброто решение, ако съществува решение.

1. Избройте предимства и недостатъци на BFS.
   1. **BFS търси в широчина. Ако съществува решение, BFS със сигурност ще го намери. Ако съществуват няколко решения, BFS ще намери най-оптималното решение. BFS използва памет, пропорционална на броя върхове във всяко ниво, което означава, че използва много памет.**
2. Избройте предимства и недостатъци на DFS.
   1. **DFS търси в дълбочина. Ако търсим в дърво с безкраен брой състояния, DFS може да не приключи. DFS може да не намери решение. DFS използва памет, пропорционална на дълбочината, на която търси, спрямо началното състояние, което означава, че използва малко памет.**
3. Избройте предимства и недостатъци на IDS.
   1. **Предимството е че, използва DFS, което използва малко памет. Също така, модифицира DFS да бъде оптимален. Недостатъкът е, че времето, необходимо за достигане на целта е експоненциално.**
4. Какви са основните типове дъги в семантична мрежа? Дайте по един пример за всеки един от изброените типове.
   1. **тип подмножество (описват релации от тип клас - суперклас)**
   2. **тип елемент (описват релации от тип обект - клас)**
   3. **тип функция (описват свойствата на обектите и класовете)**
5. От какво зависи ефективността при извличане на информация?
   1. **езика**
   2. **стила, жанра и предметната област на текста**
   3. **вида сценарий, от който се интересува потребителят**
6. Избройте основните компоненти на експертна система.

* **User interface (потребителски интерфейс)**
* **Explanation facility (обяснение)**
* **Knowledge base (база от знания)**
* **Working memory (работна памет)**
* **Inference engine (машина за изводи)**
* **Agenda (бележник)**
* **Knowledge acquisition facility (съоръжение за придобиване на знания)**

1. Кои са основните етапи на процеса на разпознаване на образи?

* **отделяне на признаци**
* **описание на обектите**
* **обучение на системата за разпознаване на образите, на базата на конкретни критерии и знание**
* **етап на разпознаване**

12. Обяснете как работи алчното (Greedy) търсене

**Greedy търсене работи като избира най-доброто следващо състояние чрез някаква евристична функция. Това обикновено прави алгоритъма бърз, но не оптимален.**

2323. Прилики и разлики между Greedy Search и A\*

**И двата алгоритъма разчитат на някаква евристика, но Greedy Search разчита на евристика, която взима в предвид само текущото състояние, докато А\* използва и цената/дистанцията до текущото състояние за евристиката. Това прави A\* оптимален, а Greedy search - неоптимален.**

122. Избройте предимства и недостатъци на Търсене в лъч (Beam Search)

**Предимството е че е Beam Search намира решение бързо и използва малко памет. Това е, защото не трябва да съхранява всички последователни състояния в опашката. Недостатък е, че не е пълен, т.е. може и да не открие решение. Също така не е оптимален - ако върне решение, може да не е най-доброто.**

12. Избройте предимства и недостатъци на Hill Climbing

**Предимства е че се изпълнява за линейно време и е константа сложност по памет. Недостатъци не е пълен и оптимален. Може да забие в локални екстремуми**

12. Какво е частично нареден план?

**Всеки алгоритъм за планиране, който може да постави две действия в един план без да има наредба кой да е пръв, е частично нареден план.**

Кога правия извод е по-добър от обратния извод при използване на системи от правила и факти? Дайте кратък, но съдържателен отговор.

**Правият (forward chaining) извод е познат още като извод, управляван от данните (data-driven inference).**

**Обратният (backward chaining) извод е познат още като извод, управляван от целите (goal-driven inference).**

**Правият извод е по-добър от обратния в началото на изводния процес, когато входната информация води до някакво бързо междинно заключение.**

Колко типа неврони има в една невронна мрежа (в общия случай)?

**В невронната мрежа обикновено винаги съществуват входен и изходен слой от неврони, във входния се въвежда информацията към мрежата, след това сигналите от входните неврони преминават през един или няколко слоя от междинни (скрити) неврони, според топологията на невронната мрежа, като сигналите накрая стигат до изходния слой, откъдето се чете получената информация.**

Какви типове извод се използват при бейсови мрежи?

* **Диагностика - от следствието към причината**
* **Предсказване - от причината към следствието**
* **Междупричинен извод - между причините за дадено следствие**
* **Смесен извод - комбинация от горните три**

Избройте основните типове алгоритми при решаване на задачи с удовлетворяване на ограничения.

* **Backtracking**
* **Constraint Propagation**
* **Local Search**
* **Forward chaining**