1. Задача и представяне то й. Подзад. Мет на пълното изчерпв Осн. х-ки при търсене на реш.

ЗАДАЧА = проблем - наблюд състоян в предм област – domain (ПО) се различ от желаното. Зад се реш чрез преобраз на състоянията на ПО по начин че разл да изчезне. T=(S,G,A). S - множ нач услов; G- цел услов (опис кр рез); A множ действ които прил в опр послед към изх условия ще довед до желания кр рез

Репродуктивни (предварително е известна послед от действ - алгорит) и ителектни задачи (послед от ел действия – алгор не е извест и се ген в процеса на самото реш), не са известни аналит. Модели (у-ния) на осн на к да се изград алгоритъм за реш – ИИ се заним с този тип. Знания са нужни. Специф на ИС: работа с знания, възмож за промяна на специк за обраб на инф.

ПРЕДСТАВЯНЕ НА ЗАД по подх начин: преди да се присъпи към решаване й. Обобщена продукционa система (формализъм за процес на търсене чрез знанията – Н.Нилсон) посредством правила/у-ия АКО у-ие, ТО => действ.

3 блока: ГБД (глоб БД – структура, матрица от числа, знакови низове); ПП (прод правила); ИНТ (интерпретатор)- модул извод. (схема нач състояние на ГБД..) ПП се прилагат към ГБД и предизв определени изменения в нея. Едно от ПП е целево условие (ЦУ) опред достигането до зададен цел. ИНТ определя кои правила да се приложат в съответ с заложената стратегия за управл (СУ) в ИНТ. Ако е дост. ЦУ се прекратява избир на правила. Ако има няколко условия – конфлик решава кое.

Редът на изпълн на правилата НЕ е предвар ясен и зависи от състоянието на ГБД по време на изпл. ОПС – програма с разкл (възмож действия на всяка стъпка. ВИДОВЕ: с право проследяване (търсене в посока към целта с прави правила); обратно (от целта към началното състояние с обратни правила) и двупосочно. За реш на 1 зад трябва да се формулира в термина на избр 3 компонента =>

ПРЕДСТАВЯНЕ на зад. 4 ПОДХОДа:

Предст чрез логически системи, пространство на състоян, простр на зад (редукц подх – свеждане/разбив на зад на подзад), и комбинирано.

ЛСис = зад се предс като съвок от лог формул и реш се свежда до търсене на *извод/опроверж* да показв че ЦУ е лог следств на нач съст ГБД е съвок от правила и факти – лог прогрма. Комб с др. Подходи. В ИИ обаче ГБД е инф с-тра не оргранiз в лог форма

ПС – пространство състоян съвок от данни за опред величини х-щи зад в даден момент време. ∀ възм съст = простр на съст (ПС). Операторите са възм дейст преобр 1 съст в друго. Цели – съст при к опер се прилага

За предс на зад в простр се използва теория на графите – *граф на състоянията (отделн съст като възли в ориентиран граф а операторите са дъгите)* - от родители и дъщерни по ниво.

Граф е геом фиг състоящ се от множество непрекъснати и самонепресичащи се криви, а ребра на графа – дъга. Ако има стрелка => направление) => ориентиран. Дървото е ориентиран граф във всеки възел влиза само една дъга с изкл нач възел. (дъщерни и родителски). Търсене на реш на зад се свежда до откриване на път в орient (семантичен) граф м/у нач и зададен *целеви възел* (възел който съдържа целево/кр състояние). Премест на дискове… 3 различни размера диска от най ляво на 1 кол и 2 свободни кола. С оператори местене от всеки до всеки.

Разб зад на подзадач - Преместв на най големи диск на последния кол е подзадача. ГБД съдържа опис на нач зад. На осн на зададени правила зад се разбива докато се стигне до финалните *елем подзадачи* реш на които е известен. Всяка подзадача се поместват в ГБД и се разбива на под-задачи. *Обед решенията* ел. подзад в съотв с стр-ра на взимоовръзк м/у тях дава решн изх. зад. И-структура (всички под нея трябва да се рашат), ИЛИ-стр (поне една от подзадачите ако е решима), И-ИЛИ-стр. (разбива се на И и ИЛИ) взимовр между подзад. Изп се И-ИЛИ графи (редукц графи) с възли. Дъгите между възлите на 1 ниво са правилата за редуцир на подзад. 1во, 2ро и т.н. ниво на възли.

Връзката м/у два звена на 1во ниво има ***"И"*** връзка =>с напречна линия се покзва.

Заключителни възли – онези които отговорят на елемент зад. (решими). Решими и нерешими възли. А онези възли които нямат дъщерни и не са заключителни са *глухи*. Дай пример с 3 канибалите, 3 мисионерите и 1 лодка!!! Почваш от корен: 0,0,0 и гониш 3,3,1 като 2,2,1 е ilegal state. С 5 налични оператора.

Простр на подзад. – множест. Съдържа описан на нач. и нейните подзад. се нар. Простр на задач. Търсенето се свежда до търсене в редукц граф на такъв подграф, вкл. Възел на описан. На нач задач позвол. Този възел да бъде маркира кат решим. Подходи за разбиряването на проблема подцели:

А/ От горе на долу - 1 разбиване на осн проблем в по-малки цели и след това рекурсивно разбиване на тези цели на по-малки цели и т.н., докато се достигнат листови възли или успешни възли, които могат бъде решен.

Б/ От долу на горе - 1 опред всички подцели необходими за решав на целия проблем и след това да започнем от решаването на проблема с успешните възли докато не бъде намерено цялостно реше.

Х-КИ при търсен на реш. Крит. за оценка: пълнота (гар. Стратегията нам. На 1 реш поне ако има?), времева сложност, пространст сложност (колко памет е нужна), оптималност (гарн стратегията най качественото реш. с критерий за качество) зависи от обема знания налични или пълна неинформ. Ефик. На търсене се харак с изчислит загуби: загуба от прилагане на правила и от избор на правила.

ВИДОВЕ стратегии за търсене: А/ Data-driven /orward chaining search - starts from an initial state (the root node in the search tree) and uses actions that are allowed to move forward until a goal is reached. Б/ can start at the goal and work back toward a start state, by seeing what moves could have led to the goal state - also known as backward chaining.

А/ са: *Сляпо пълно изчерпване (неинфор)* Търсене когато не се знае нищо (no additional knowledge) за зад. и е кр неефективно. Зададена предвар твърда схема на обхождане (форм. на пътя на реш). В дълбочина, широчина и комбинирано.

Насочено (евристич) търсене – възможн да се фокусира и редуцира търсенето (отсичане на част от графа на състоянията) с интуитивни и експетни знания за конкр. Задача водещи до повиш ефектив. търсене при известен риск. Критерий показващ до колко възела е добър от останалите. Точни но не винаги се достига до оптим решение или въобще реш.

2. Метод на изкачван. Градиентен мет на най-бързото изкачван. Търсене на цел при ограничи у-ия

Метод на най-доброто изкачване

На всяка стъпка да се избира най-добрият от всички генерирани до момента наследници и той става текущ.

Ако не сме достигнали целта оценяваме тези наследници и ги добавяме към общия списък наследници -> първото

Често се ползват методи които позволяват да не се генер целяото ПС и може да НЕ се стигне до реш. 1 често ср случа на информирано търсене е случаят, когато в пространството за търсене може да се построи *оценъчна функция* с която опреде степентта на близост на даденето състояние до търсената цел. Метода се нарича *изкачване* при положение и когато оценяваща функция *нараства* с приближаване към целево състояния.

МЕТОД НА НАЙ-БЪРЗОТО ИЗКАЧВАНЕ (HILL CLIMBING) - Simple модел. Алгоритъма е цикъл докато не се намери реш. (целво състояние) или се изчерпят възмож за генериран на наследници. Генер нов наследник на тек. Състояния => оценява се

новото състояние ако не е целево но е по-добро (а може да има и др. Много по добро) от текущото то става текущо. Ако не е по добро то цикъла продължава. Пътя от възли до целта се записва. Няма опция за възврат търсенето е еднопосочно.

I стъпка: Оценяваме началното състояние ако то е целево алгоритъма спира. Ако не е целево то го обявяваме за текущо.

II стъпка: Цикъл от следните действия които се повтарят докато се намери решение или

А) Генериране на наследник;Б)Проверяваме дали този наследник е целево състояние => край

В) Ако не е целево преценяваме наследника

Г) Ако оценката е по-добра от текущо, то става основно; Д) Ако оценката на наследника е по-лоша цикъла започва със същото състояние; Е) Ако се изчерпят възможности за генериране на наследници на тек. състояние се връщаме назад и текущо става предишното състояние

____

Ако се разгледат и оценят всички възможни наследници (а не само най блзкия с по добра) и този с най-високата оценка се номинира и се сравнява с текущото – това е ГРАДИЕНТЕН МЕТ. НА НАЙ-БЪРЗОТО ИЗКАЧВАНЕ. (Steepest ascent hill climbing). винаги проверявате около вас във всичките четири посоки и избирате най-висока позиция.

Алгоритъма спира ако наследника с най-добра оценка измежду всички *наследници* не е с по добра оценка от *текущото състояние*. Алгоритъма с цикъла спира когато текущ състояние съвпада с целевото. Не винаги водят до намиране на реш ако се дост до съст което няма наследници по добри от него

Х-ки на 2та метода: избора на оценяващата функция е много важна. Не винаги водят до достигане на *състояние* което е *целево* и което няма наследници по дорби от него. Възможни проблеми: *локален екстремум* – състоянието е по-добро от наследниците си, но не е най-доброто в цялото ПС; *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.

ТЪРСЕНЕ НА ЦЕЛ ПРИ ОГРАНИЧИ У-ИЯ

- Constraint satisfaction

Да се построи описание на търсеното целево със то, което удовлетв дадено множество от ограничит условия. Прим задачата за осемте царици, решаване на *криптограми SEND + MORE = MONEY10 на 40 възмодфд*, съставяне на *разписания*, различни проектантски задачи (с различни ограничения откъм време, пространст, цена и пр.). Алгоритъм за намиране на цел при ограничителни условия - Първоначалнот множест от ограничител услов се разширява, като в него се включ и ограниченията, които са логически следствия от вх. Този процес се нарича *разпространяване на ограничител условия*. След това, ако не е намерено решение, се прави предположение (хипотеза) за някой от неуточнените параметри. По такъв начин ограничителните условия се засилват, след което новополучените огранич условия отново се разпространяват и т. н. Целта е да се достигне до противоречие (тогава се осъществява връщане назад и се прави ново предположение за съответния параметър, ако това е възможно, и т. н.) или до намиране на решение - множеството възм ст-сти стане празно. А при достигане на противоречие се предприема връщане назад с отказ от последни действия и се формулира нова хипотеза и т.н. За да може да се осъществи търсенето при този метод, необходимо е да са известни: а) правилата за разпространяване на ограниченията; б) правила, по които могат да се генерират хипотези.

1. Прави се списък (О) на всички обекти, на които целевото състояние изисква да се присвоят конкретни стойности (начални ограничителни условия). Разпространя се началните ограничите условия.

2. Изпълняват се следните действия, докато се стигне до противоречие или списъкът О стане празен.

а) избира се обект от О; разпространяват се ограниченията, които засягат този обект;

б) ако така получените ограничения са различни от ограниченията при предишното разглеждане на този обект (или той се разглежда за първи път), към О се добавят всички обекти, които участвуват заедно с разглеждания обект в описанието на някакви ограничителни условия;

в)разглеждания обект се отстранява от списъка О и се записва като последен в списък Р.

3.Ако обединението на получените ограничения удовлетворява решението следва край. В противен случай се прави връщане към някой от обектите в Р, като всички обекти от края на списък Р до този се преместят в списък О и от обединени на получени ограничен се отстранят онези ограничения, които са били следствие от обработката на преместе обекти.

4.Процедурата се повтаря докато се изчерпят всички възможни хипотези.

3. Игри

Играта е естествен начин за обучение. Игрите най често реш задача за намиране на най-добър ход на играча, който трябва да направи. Игрите се класифиц: пълна информация (observable – пълна инф за хода на играта) и непълна информация при у-ия детерминистич и (не) случайни – шанс като таблата. При игрите с непълна информация целим да получим повече информация, и не винаги толкова по-голяма оценка.

При игрите има *непредсказуем* опонент (чиито цели са противоположни на нашите) и времеви ограничения. Коеф на разклоняване – среден брой коректни ходове в 1 поз. Дълбочина са бр възли (100). Състояния при която играта свършва са терминални. -1 загуба, 0 реми, 1 победа. Дървото на търсне е разделено на нива като на всяко инициатив принадлежи на 1 играч.

ИГРОВИ СТРАТЕГИИ

А/ МИНИМАКСЕН алгоритъм. Изисква се построяване на цялото ДС и намиране на оценките на листата (статични оценки). Първи играч е максимизиращия играч. Метод за получаване на оценките (придобитеи или породени) на възлите от по-горните нива на ДС, които позволяят на 1вия да избера най-добрия си ход – предполага се че имаме пълното дърво на играта до термин възли. С двама играчи всеки се стреми да *максимизира* своята оценка/поз състояние и да *минимизира* тази на противника. Придобити/ породени оценки. Една задача е: complete - ако дървото е определено (небезкрайно). Пример: шах и Optimal - ако се играе срещу оптимален (възможно най-добър) противник. Оценките се разпр *от долу на горе* започвайки от терминалите като всеки от възлите съответни на ход на максимизиращия играч избира оценка, равна на максимален от оценките на преките му наследници, а весеки от възлите, съответни на ход на минимиз играч, се избира мин. оценка на неговите наследници.

Б/ ЕВРИСТИЧНО оценяване на състояние - Дървото се отсича от определен възел надоли и се намира еврист оцен на състоянието. Предполага се че при дадене позиция Макс ще победи а Мини ще загуби. Ползва се опита на добрите играчи. Разбира се евристиката може да се усъвърш експериментално. При шаха се прави оценка на броя фигури на база 1 пешка – 1 кон = 3 пеш а разполож на фиг. Да се отчита с доп коеф.

Неефективен но точен и за шах не е добре. Ген на дървото е напълно отделен проц от оценява на позициите.

В/ АЛФА-БЕТА. Стратег съкращава бр перспективни възли – кастрене остичане алфа – бета (cut-off). Започва се от лявите листа на цяловото дърво. Вкарават се 2 параметъра по време на търсене в дървото. *Алфа*– стойност най-голямата получеа досега стойност на възлите на Макс и бета – наймалката получена стойност на възлитена Мини. Ако възела на Мини има ст >= на *бета* на неговите съседи на същото ниво могат да се пренебрегнат (не се извършва генериране

върху всяко поддърво) и обратно ако възел на Мак има стойн по <= на *алва* неговите съседи на също ниво могат да се пренебрег. Но ефектив зависи от *реда* на обхождане на наследниците на позициите в дървото. Ако най-перспек ходов се разглеждат първи се намалая сложността по време. Листата се оценява веднага след генерирането й. Намаля скоростта на развитие на мобинарония взрив но не го предотвратява. Да се определи нивото на отсичане.

В/ Игри с случаен елемен. – табла. Освен нивата на дървото се добавя и нива на случая – заровете които състояния зависят от възможните изхода)

4. Стратегии за сляпо търсене в ПС. Методи и сравнение. Както вече казахме сляпото търсене е генериране на цялото дърво с цел изследване без да имаме доп. Информация. Обхождана може да става в:

А/ *проби и грешки* – случайно прилагане на някой опертор докато евентуално целта не бъде достигната. Не гарантира намиране на реш.

Б/ Метод на търсене в *Дълбочина* – depth-first search обхожда се даден клон (по 1 път се върви до края му) и като се стигне до края му без решение се преминава към следващия най-близък (възврат до най-близкото разклонени) с необходени наследници. Създават се т.н списъци с изследвани О и неизслед състоян С - множеств открити възли – кадидати за разкриване). И маркери които покзват текущо състояние. Решение представлява последовате от операторите съотв на ребрата свързващи възлите в посоката от старт. До целевия възел. По малко памет. Ако има много реш ще е по бърз от Широч. Непълен метод. Ако се тръгне по безкраен път без целеви възел няма да има решение. НЕ винаги намира най-краткия път. Времево е експоненциално. Трябва да се избягва при клонове с голяма и безкр дълбочина. Време = b (фактор на разклонение на дървото) на степен m (макс дълбочината на дървото); Памет = b*m
Пълнота (гаранция нам реш): НЕ,
Оптималност: НЕ

В/ метод търсене в *Широчина* – последова се обх всички възли на едно равнище като се започне от корена и се преминине на следващо ниво. Пр от ляво на дясно. Винаги се намира решение. И то най късия път защото върви по редове. Експоненциален х-р са максималния бр на разкр възли. Нужна е много памет. При случаи както е с шаха и Go с много разклонения не се справя добре. Или ако има много разклонения с еднаква дължина завършващи с целево състояние. При търсене в дълбочина новопородените възли се добавят в началото на списъка а при търсене в широчина в края, което опред реда на обхождане.
Време/памет= b (фактор на разклонение на дървото) на степен d (дълбочината на реш);
Пълнота (гаранция нам реш): ДА,
Оптималност равни цени
В/ *Комбинирано* – интеративно задълбочав (Iterative Deepening Search or IDS): Започва се с дълбочина и се установява гранична дълбочина след което търсенето продължава по новия път докато се достигне отново гранич дълбочина. Възлите намиращи се на гран дълбочина не се разкриват. Х-ра се: пълен – гарантира намиран на реш ако съществ. Най-краткото, някои възли се разкр многократно.
Време = b на степен d;
Памет = b*d
Пълнота: ДА
Оптималност: равни
Г/ Метод за търсене на оптималния път – когато имаме оценяващ критерий с параметри като цена на дъгата или цена на прехода отговарящ на стойността на критерия. Не просто намиране на решение а на оптимално – оценките от които да е най-малката сума. Път с мин цена. От изследване на онези възли които имат мин. Цена. Редуцира бр възли за разглеждане! Ако дъгите на всички възли имат една стойност този метод е = на широчина.
Време = b на степен d;
Памет = b на степен d
Пълнота: ДА,
Оптималност: ДА
Д/ Двупосочно търсене – напред от начално ст-ние и назад от целвовото докато се достигне общо състояние което свързва двете. Назад предполага предшествениците да са обратими. Какъв метод ще се използва за 2те посоки. Време / памет = b на степен d/2;
Пълнота: ДА,
Оптималност: ДА

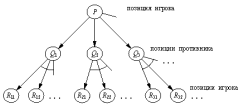
Сравнение на методите по: Време/ Сложност и Памет са сходни параметри; Пълнота и Оптималност.

5. Роля на знанията в процеса на търсене. Стратег за насочено търсене в ПС. Както стана ясно търсене в дълб и широч се прави когато няма доп инфор за дървото което се обхожда (сляпо). Подобно на откриването на изход в мазе ако се държи лявата стана на стената. Но в много случаи такава информация съществува – знание (осн на опита) и може да се използва при търсенето. Нарича се евристично и човек много често го ползва в практиката. В осн на евристичното търсене стои идеята за сортиране на кандидатите възли за обработка в съответстви с някаква мяра, оецнъчна функция (heuristic evaluation function). Алгортмите за такъв подход се наричат алгорит за авристично насочено търсене. Осн идея на еврист методи се състои в сортиране на О (кандидатите за възли за обработка – разкриване). Съществ различни подходи за построяване на оцен функц. – опр разстоянието/степен на различие между дедения и целта или вероятност дадения възел да лежи в/у оптималния път – най краткия/с нисък разход до целта.
=> избирането на правилна евристика (функция) на база опит и интуиция на изследоватяля е кл момент при тези търсения. 1 Метода се нарича *изкачване* ако оцен ф-ия *нараства* към целевото състояние и *спускане* ако *намалява*. Виж. Въпрос 2 за методите Hill Climbing (отисичане на на графа) използва се функция която оценява степента на близост до целта).
О е списък за разкриване, С списък с вече разкрити.
В/ Метод на най-доброто спускане (Best-first search) – е евристичен но вариант на пълното ичерпване. Тук евристиката е свързана с избор на посока (оценяване на разхода за стигане на целта) е не с отсичане. Започва в дълбочина от най-перспективния (частично се разкриват), ако в дълбочина (едно ниво по надолу) не се намери реш или са по високи оценките се връща на изоставен такъв. Към спсъка се добавят разктити възли с оценките и ако 1 ниво по надолу е с по високи от от тези по горно се разкриват и другите от по горното!

Г/ А*-алгоритми, който вкл. Като евристика оптималния път = онзи път с най ниска цена и касае най прекия път от началото до целта. Горните методи може и да саа го лучкали само по случайно но ори тях няма гаранция че ще го намерят.
Търсене с минимизиране на общата цена на пътя (А*) - 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 ГС са краен брой и цената на преходите е положителна, и оптимален, ако евристика е приемлива (оптимистична), т.е. никога не надценява стойността (цената) h* на оставащия път (ако h*(node) ¥ h(node) за всеки възел node).
Цената на оптималния път path-based evaluation function (оценъчната – планирана функция) f*(v) = g*(v) (от някой произволен нач. възел до текущия възел) + h*(v) (от текущия до целта – трябва да е подценана). работим с тяхните оценки.
За 2 нода m and n, и heuristic function f, ако f(m) < f(n), => се очква m да попада в *оптималния път* за достигане до целвият възел от колкото n. f(m) е по информира евристика. Оперира като алг за най бързото спускане но претегля възлите на база оценъчната функция f*(v) – по горе описана. Само когато h(node) е подценена

се гарантира оптималност – най-късия път. Ако евристиката не е допустима (не е underestimate) не се гарантира най късия път => алгоритъма се нар. “А”. Или h*(v)<= h(v) за да имаме подценяване.
Под алгоритми са:
- Uniform Cost Search открит и наречен: Dijkstra algorithm където h(v) is set to zero и се гледа сравнява само g(v). трябва да се продължи след като се намери реш за др. по добро…
- Greedy Search - g(v) = 0 и се сравнява само h(v). Може да не открие решение.
Обща характеристика: реализират пълно изчерпване по гъвкава стратегия или търсене с отсичане на част от графа на състоянията. Х-ка: Приложими са при наличие на специфична информация за предметна 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.

6. Варианти за форм зад за търсене в/у граф.
СПИЦИФИКА НА ТЪРСЕНЕТО – *търсене чрез разбиване задачата на подзадачи* - използв на редукц подход с И-ИЛИ граф. Този подход на реш на по малките проблеми се нарича goal reduction а дървото е И-ИЛИ. Някои може да се решат само с решаването на всички подзадачи – И-възли, които представят И-целите а др. целта може да се достигне ако се достигне до някоя (поне една) от подцелите – ИЛИ-цели. *Основната разлика между целевите дървета (И-ИЛИ графите) и нормалните дървета (граф на състоянията) за търсене е това, че за да се реши проблем с използването на целево дърво, редица подпроблем (в някои случаи всички подпробле) трябва да бъдат решени, за да се реши основният проблем.* Следователно, листовите възли се наричат *успешни възли*, а не целеви възли, защото всеки възел представлява успех в малка част от общия проблем. Ациклични графи в които няма цикли. Цикъл е път който тръгв от даден възел и свършва пак в него. Ациклични – И-ИЛи в които не съществ възел притежаващ потомци, които да се едноврем и негови предшественици. Примерна задача е да се преместят n дискаот стълб I през M върху P. Z = (n, I,M,P). Преместването на 1 диск (n=1) (1 диск се мести) се счита за елементарна задача. А реш са 3 подзадачи (*пространст на задачите*). Търсене в редукционен граф на такъв подграф който вкл възела на описанието на нач. Задача и позволява да бъде маркиран като решим. И този подграф представл реш на задачата. Игрите като шах може да се представят с граф И/ИЛИ.

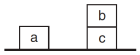


Осбености: а/ ПС един оператор може да породи само 1 дъщерен възел то при И-ИЛИ предств моеже оператора да породи множество дъщерни възли; б/ търсене на реш с И-ИЛИ графи се осн на обратното проследяване- започва се от възела представляващ целта и се цели генерир на множ зад; Целта на търсенето при редукционното представяне се свежда до генериране на граф на решение с постепенно изграждане.
МЕТОД НА ТЪСЕНЕ В ДЪЛБОЧИНА В И-ИЛИ ГРАФ
Алгоритъм (цикъл): 1. O← v0 (запис в списъка O на частичен граф състоящ нач възел V0), 2. O трябва да съдържа реш иначе СТОП - НЕ, Избира се 1 ел – частичен граф от O и се означава G, ako G решим => Стоп –ДА, Генериране на подграфите чрез разкриване на кр. Възли на G с маркиране и оценяването и записване в нач. На на списъка O, Преход към 2. Търсене с възврат е модифицирано на горния алгор
МЕТОД НА ТЪРСЕНЕ НА ОПТИМАЛНОТО РЕШЕНИЕ В И-ИЛИ. *Хипер* Графи (хипер дъги) свързват родителс възел с множество дъщерни възли. Бр на това множество определя ранга на хипердъгата - k. k-връзка. Търси се цената решаващия граф чрез цена на всяка връзка = оптимален по зададен критерий. Цели се ефект да се повиши чрез използв евристична ф-ия по подобие на А* алгоритаъма и се нарича АО* - предполга че за всеки възел на даден И-ИЛИ граф е зададена или да се пресметне ст-ста на евристичната фукжия h*(v), която предств *оценка* на ф-ция h(v), опред цената на оптим решаваш граф, свързв възели V с множеств закл възли. Ако h*(v) = 0 става алгор за търсене в дълбочина. И при АО* може да се наложи *огранич за монотонност*

7. Избор на метод за търсене. Методи за търсене в големи пространства.
А. ИЗБОР НА МЕТОД ЗА ТЪРСЕНЕ.
Изборът на метод за търсене зависи от:
- размерността на пространството на състоянията на системата;
- изменяемостта на проблемната област;
- пълнотата и дълбочината на модела на
областта;
- определеността на данните на решаваната задача.
- броя на търсените решения (едно, няколко, всички);
- известните или предполагаеми ограниче за резултата.
Ще приведем една възможна класификация на методите за търсене в ЕС:
1. Методи за търсене при неголямо пространство, статична област, пълен модел, точни и пълни данни.
2. Методи за търсене в големи пространства с йерархична структура при статична област.
3.Методи за търсене в пространство на неточни и непълни данни.
4.Методи за търсене в динамични изменящи се във времето и/или пространство проблемни области.
5.Методи за търсене в сложни бази знания, за представянето на които се използват няколко различни модела.
Б/ МЕТОДИ ЗА ТЪРСЕНЕ В ГОЛЕМИ ПРОСТРАНСТВА.
Търсенето в големи пространства често изисква по-специал техники, за да се постигне резултат за приемливо
време. Ще спомена накратко някои от тях.
1) Намаляване на пространството чрез приемане за фиксирани параметрите, за които не е указано, че се изменят в конкретния случай (по подразбиране) или които се изменят с много по-малки скорости.
2) Разделяне на пространството за търсене на отделни подпространст
1. търсене на необходимото подпростр (в което е решението)
2. търсене в отделеното подпространство (търсене на самото решение).
3) Метод на абстрахирането: Пример: път от центъра на град А до центъра на град Б.
4) Метод на генериране и проверка
а. при зададени конкретни условия, ограничения и други данни се генерира предполагаемо решение;
ж. генерираното решение се подлага на проверка.
5) Използване на метазнания.

8. Планиране. Основни методи.
А. СЪЩНОСТ - Подход, при който се генер последователност от елем действия за изпълнение на задача или за постигане на определена цел, се нарича планиране. Представянето на даден проблем като съвкупност от *състояния (как изглежда средата в момента)*, *действия (описание на средствата)* и *цели (как би трябвало да изглежда средата)*. В класич среди за планиране – *агентът* (planer) извършва крайни, позволени и конкретни дискретни елементарни (atomic action) действия. Кл момент е намирането на функция на добра евристика и как да се декомпозира и изпълни на части проблема в повечето случаи като дърво с подцели. Ген *план* представл линейно или частично нареда послед от *оператори* за вески който е описан *модел* на допуст действие. Разлика м/у търсене (действ се разглеждат като черна кутия, избор в резултат на евристич ф-ия => избора е проблем) и планиране (съст и целите се описват като изречения и има логически описание при което съст => някакво действие, позлва се метода на разбиване на задач на подзадачи)
Б/ Преставяне на съст, целите и действията – на

език за класическо планиране език STRIPS. *Състоянията*- чрез литерали които не съдържат функц. Имена (променливи) или Приема се, че светът е затворен - това означава, че всички условия, които не се срещат в състояние се считат за лъжа; Целите – може да съдържат променливи; Действията чрез оператори – индентификатор, предисловия, отпадащи литерали и добавяни литерали. Действие = Предусловие (положител литерали, посочващи какво трябва да бъде изпълнено в състоянието преди да може да се изпълнени действието) + Ефект (конюнкц на функционално-свободни литерали, описващи как се изменя състояни, когато действието бъде изпълнено. Операторa е приложим ако са изпълнени предусловията. Наричаме схема на действие. STRIPS in action, in the blocks world, which consists of a table, three blocks (a, b, and c) and a robot arm that can move blocks around. two predicates to describe the world: *On (x, y)* means that block x is on top of block y & *Clear (x*) means that block x has no block on top of it. On (a, t) means that block a is on the table. *Clear (t)* will always be true because we assume that the table is large enough to hold at least three blocks at once. Our goal is to place block c on top of block a, which can be stated as On (c, a). start state can On (a, t), On (b, c), On (c, t), Clear (b), Clear (a), Clear (t). Operator schema: MoveOnto (x,y), which means “move object x from wherever it is, and place it on top of object y.”



В/ ОСН. МЕТОДИ за Т в ПС – или т.н. планирането на алгоритми. ПОДХОД свързан с езика STRIPS и ИЗПОЛЗВА ТЪРСЕНЕ В ПРОСТР ОТ СЪСТОЯНИ И простр на ПЛАНОВЕТЕ. Чрез насочен граф, възлите в който преставл състоянията а дъгите – операторит в STRIPS. Тъй като описанията на действията определят както предусловията, така и ефектите, е възможно търсенето да се осъществи в една от двете посоки:

- *Планиране чрез прогресия на Действието* -търсене напред от началното състояние. Разглежда ефекта от всички възмож действия в дадено състояние. И кое трябва да се приложи. Има голяма възможност да се сбърка при висок фактор разклонения защото има голямо простр на търсенето. Добрата евристика е задължител за постигането на ефективно търсене. За начално състояние при търсенето ще считаме началното състояние на проблема. Като цяло, всяко състояние е множество от положите основни литерали. Литерали, които не се споменават са лъжа. Действията, които са приложими в състояние, са всички онези чиито предусловия са изпълнен – на принципа: добавяне на положителни ефекти и изтриване на отрицател.

-*Планиране чрез регресия на Целта* - търсене назад от целта. За достигане на целта, това което трябва да бъде вярно е предишното състояние. Каква е следващата подцел да се удовлетвори. Предпочит метод. Но как да определим предшестве и кои са състоянията, които след прилагане на някакво действие биха водили до целта? В допълнение към изискването на действията да достигнат до някакъв желан литерал, трябва да се уверим, че тези действия не премахват всички желани литерали. Действ, което удовлет тези ограничен, се нарича *консистентно или последователно*. Главното предимство на този алгоритъм е, че предприетите действия са винаги уместни. Често факторът на разклонение е много по-малък, отколкото при алгоритъм за планиране чрез прогресия. Основната идея е да се изчисли колко действия биха били необходими до достигането на целта – правилната евристика нужна за всеки от посоч по горе 2 метода. Търсенето може да се осъществи чрез всеки стандартен алгоритъм за търсене. Прекратяваме процеса, когато е генерирано описание на предшественик, което се удовлетворява от начaлн състояние на проблема.

Др. подход е търсене в простр на плановете (GraphPlan) – планиране, като се започне от първоначалното състояние и се работи към целта чрез извличане плановете. Графиката на планиране се състои от няколко нива. Първото ниво (на нулевo) съдържа предложенията които са верни в началното състояние на проблема. Следващото ниво на графиката съдържа действията, които могат да бъдат извършени в това състояние. Нивото след това съдържа държавите, които могат да бъдат доведени чрез извършването на тези действия. Следователно всяко равномерно ниво в плана представлява състояние и всеки нечетно число представл действия. Крайното състояние в графиката представлява целта. Algoritym: Първо, предложенията, които описват целта, се сравняват с сегашно състояние. Ако всички тези предложения са налице и няма два от тях са свързани с mutex връзка, тогава е възможно, че решение вече е е достигнато. На този етап се изпълнява втора фаза на алгоритъма, за да се опита извлечете план от настоящия графичен план. Ако текущото състояние не съдържа всички необходими предложения, тогава следващото ниво на графиката за планиране се получава чрез прилагане на всички приложими операторите и да определят всички възможни предложения, които могат да бъдат изпълнени от тези оператори. Този алгоритъм се повтаря, докато не бъде намерен подходящ план или докато не може да бъде че няма план.

- *Частично наредени планове.* Досега разгледаните планирани (прогресивно и регресив) са видове пълно наредени планирания. Можем да изградим пълно нареден план, работейки с частично наредени планове, Планът, който изграждаме, включва две независими последоват от действия примерно обуването на чорап за ляв и десен крак. За разлика от пълните наредби, тук не заявяваме строг ред на изпълнение на тези последователности.

Планирането започва с непълен частичен план. 1 частичен може да се разглежда като множеств от пълни, всеки от който изпълня множество от ограниче. И се използва 2 вида оператори – прецезиран и модифиц. Всеки алгоритъм за планиране, в който могат да се поставят две действия в план, без да се уточнява кое се случва първо, се нарича *алгоритъм за частично наредени планове*. Реше е представено във вид на графика на действията, а не като таката тяхна последоват. Състоянията са, най-често незавърше, планове. Празният план съдържа единствено начал и финалното действие. Всеки план се състои от 4 компоненти:

*Множество от действия – това са стъпките на плана.

*Множество от наредени ограничения: A<B. Циклите представляват противоречия.

*Множество от причинно-следствени връзки: A--p-->B

*Множество от отворени предусловия – ако предусловието не е достигнато от някое действ на плана.

С частично нареденият план действаме по следния начин:

1. Избираме едно отворено предусловие.
2. Търсим всички консистентни планове-наследници, т.е. поредици от действия, които удовлетворяват условието.
3. Добавяме причинно-следствените връзки и ограниченията към плана (A--p-->B и A < B).
4. Разрешава конфликтите между новата причинно-следствена връзка и съществуващите планове, както и между новото състояние A и съществуващите планов.

Един план е консистентен, ако в него няма цикли в наредените ограничения, и няма конфликти в причинно-следствените връзки. В процеса на действие, частично нареденият план се преобразува в пълно нареден. Консистентен план, който не съдържа отворени предусловия е решение. Вече сме готови да формулираме задачата, която частично наредените планове решават. Ще започнем с формулиран на подходящо твърдение за проблемите на планиран.

Както обикновено, дефиницията включва началното състояние, действията и целта.
Началният план:
*съдържа състоянията Начало и Край
*ограничението Начало < Край
*не съдържа причинно-следствени връзки
*всички предусловия в Край са отворени
Функция на наследника – избира произволно едно отворено предусловие р на действие В и генерира план за наследника на всяко възможно съответствие на действието А, което удовлетворява р.
=> Тестването на целта проверява дали планът е решение на първоначално поставения проблем,
т.к. са генерирани само консистентни планове, тестването трябва да провери дали не сесъдържа отворени предпоставки.

9. Сиситеми с ИИ и информационн системи (ИС – комп с-ми). ИИ е познавателна и инженерна научна област която се занимава с разбиране и исинтезиране на пораждането от интелекта *интелигентно поведение* – се разбира взаимодейств м/у система и среда, характерно *за* което е да се *знае, разсъждава и действа целенасочено.* Включва възприемане, интерпрети на инф от средата, лог разстждения, обучение, вземанени на реш и действие. Системи с изкуствен интелект- още *интелигентни системи* или *иинтелектни агенти (ИА)*. Всяка система способна да води интелигентно поведение се нарича *интелектна*. Агент ее всяка система която може да възприема информация от външ среда посредств сензори и изпълнява действия в нея посредсв ефектори – изпълн у-ва. ИА е система , притежав способност за:
- извиличане , представяне натрупване, коригиране, интерпретиранеи общаван на знания чрез активно взимоод с средата. – използв на знания за водене на целенасочено поведение. *Реактивност*– възможностдареагира адекватно на промени в околната среда; *Целенасоченост* – способностдапредприема действия в посока на изпълнение на поставената цел
• *Адаптивност*– възможностданагажда поведението си в съответствие с измененията на условията на функциониране
• *Социалност*– възможностдасекоопери ра или да се договаря с други агенти. ЕС са част от СИИ. ЕС е програма, която въз основа на структурирани по определен начин знания решава сложни практически задачи, обосновава и обяснява своите решения, натрупва нови знания, реализира смислен диалог. Основни компоненти на една експертна система са базата данни, базата знания и програмата, която осъществява извода.
Има интерфейса на системата, който дава възможност тя да бъде създадена и да се усъвършенствува (среда за създаване и развитие), а от другата е интерфейса към ползувателите на системата. Основаната инженер цел на ИИ е изграж на интелектни системи. Интелектните системи се създават за решаване на задачи, представл практически интетер и затова борави с *знания от дадена предметна област*. СИИ моделират сложни човешки дейности и задач. СИИ се изграждат въз основа на два подхода:
- структурна аналогия с невронните мрежи (моделират структурно и функционал нервната система на човека)
- функционален анализ на разумна човешка дейност (структури от арсенала на изчислителната техника) Примери за такива задачи: разбиране и синтез на текстове на естествен език; възприем и синтез на реч; анализ, обработка и синтез на изображения; превод от един на друг естествен език; вземане на решения при непълна информация или в условията на изменящи се околни условия; автоматизирано проектиране; автоматиз създаване на планове;
КЛАСИФИКАЦИЯ НА СИИ: системи с общо предназначение и специализирани систем;
Системи с общо предназначение – създават на основата на метапроцедури за търсене процедури за решаване на различни конкретни задачи. Технология за използува им: експерт създава база от данни и база от знания за решаване на проблема;
потребителят задава конкретното приложени; при всяко конкретно използуване се формулира цел и се задават конкретните начални данни; системата генерира начина на решение и получава конкретно решение. Основните приложения на интелиге системи са в: Диагностика на пробле , Прогнозиране
РАЗЛИКА МЕЖДУ ОБИКНОВЕНА ИНФОРМАЦИОННА СИСТЕМА (ИС) И СИИ
ИС – Архитектурата на СИИ включва следните базови елементи:
1)База знания - съдържа факти и правила,свързани с предметната област; 2)Механизъм за лог. извод, действащ като управляваща структура за изп. на БЗ при търсене на реш. за конкретна зад.; 3)Глобална БД - поддържа канал за входните данни и съхранява инф. за всяка зад.; 4)Механизъм за диалога “човек-машина” с необходимото упрваление на разговора; 5)Ядро на ИИ – инструментариум или средства на ИИ (езици за програмиране и др.)
2. БЗ не е съвкупност от случайни сведения, а множество от конкретни факти, свързани в единна структура. Това е съвкупност от факти, обхващащи цялата инф. за дад. предметна област. БЗ се състои от 3 осн. компонента: 1)правила, съставени от 2 части – АКО <условие> ТО <извод>; 2) въпрос-подсказка, предоставящ възможност за запитвания относно необходимата допълнителна инф.; 3) превод, позволяващ дообясняване на ключовите думи и по-удобно възприемане на изводите.
3. ЛИ – Целта в СИИ е отделяне на данните (БД) от знанията (БЗ), а двете от управлението, т.е. модела:
=> СИИ База от знания
<u>База от данни</u>
Обработка на знания
<u>Обработка на данни</u>
Преобладаване на:
Декларативна спецификация
<u>Процедурна спецификация</u>
Дедуктивен механизъм
<u>Алгоритъм</u>
Индуктивен механизъм
база данни алгоритъм (програма) алгоритъм + данни процедурни спецефик. (създаване на алгоритъм)
СИИ – база знания, механизъм за извод знания + данни, декларативни специфик. (описания на знанията на данните)
Търси се универсален подход, който би бил приложим към всяка задача. Описваме ситуацията:
-Има обекти в някаква подредба
-Задачата се дефинира чрез задаване на крайна подредба
-Решаване на задачата изисква да могат да се пресмятат следните оператори: разпознаване на идентивност; намиране на различие; отстраняване на различие. Всяко приложение на оператори води до изменение на ситуацията и всяка задача може да бъде описана с:
1.Множество обекти
2.Списък (набор от оператори)
3.Процедура за откриване на различия
4.Таблица даваща връзката оператор
5.Начално състояние
6.Целево състояние
От 1-4 –семества еднотипни задачи
5 и 6 спецефични за всяка конкретна задача. Такъв подход има, но трябва да има две процедура (1)-преоброзуване на началната (1) и (2) ситуация в целева.
-Стъпка 1 – установява разликата между началната ситуация и целевата, ако разлика няма, следва край. В противен случай, следва стъпка 2.
-Стъпка 2 – установява най-съществената разлика и избира оператор, които може да я отстрани.
-Стъпка 3 – приложи избрания оператор към началната ситуация и новото състояние обяви за текущо.
-Стъпка 4 – премини към стъпка 1, като вместо с начално, работиш с текущото състояние.
Процедура (2) – прилагане на оператор към обект
-Стъпка 1 – изходният обект се преобразува по такъв начин, че към него може да се приложи оператор
-Стъпка 2 – прилагане на оператор. Репродукт (предварително е известна послед от действия и с

алгоритм език може да се опише) и интелектни задачи – послед от задачи не е известна и трябва да се генерира в процеса на самото решаване.

10. Механизни за разсъждения. Знания и факти. Дедукция, Индукция, Аналогия, Евристика. Съждението е твърдение за съществува или несъществуване по отношение на даден предмет или характерист. Простото съждение съдържа: логически субект - понятието, което отразява предмета в твърдението (SI); предикат - понятието, което отразява някаква характеристика на предмета (PI); логическа връзка - понятието, което отразява някакво отношен между логическия субект и предиката (R1). Ако съждението включва някаква характеристика на логическата връзка, това означава, че в структурата му присъства определен вид модалност (M,). Някои логици твърдят, че дори отсъствието на подобна характеристика е вид модалност. Нещо повече, когато модалностт не е изразена в някои съждения, това означава само, че в тях е извършена епистемологическа операция - абстрахиране от определени модални характеристики, проявени в познавателната ситуация. Основните закони като: *Закон за непротиворечивостта*. Две взаимно отричащи се твърдения за един и същ тъждествен сам на себе си предмет (предмет, спрямо който е проявен закона на тъждеството) не могат да бъдат едновременно истинни в едно разсъждение. Индукцията, дедукцията и аналогията са трите фундаментални логически процеса на логиката на съжденията (три начина на правене на логически умозаключен). Всеки от тях стои в основата на вид логическо мислене.

	Дедукция	Индукция
Описание	Тръгваме от изходни предпоставки и чрез логическите закони получаваме заключения.	Наблюдаваме обекти и феномени и правим обобщения (прехвърляне на свойства по аналогия, прогнозиране и пр.)
Надеждност	Сигурни сме в получените заключения.	Възможно е заключението да е невярно.
Обхват	Получените заключения не са по-силни от предпоставките.	Заключенията могат да бъдат по-силни от предпоставите.
Посока	От общото към частното.	От частното към общото.

ДЕДУКЦИЯ – от общо към частното разсъждение. При дедукцията, това, което е валидно за даден клас от неща, е вярно за всички елементи от този клас. Например: Всички хора са смъртни. Иван е човек. Следователно, Иван е смъртен. Оттук става ясно, че за да бъде вярно заключението, направено посредством дедуктивна логика, то изходната хипотеза трябва да е вярна. Дедукцията е процес, който работи най-добре при затворени системи, като математиката или формалната логика, където например е пределно ясно, че следното твърдение е вярно:Това е триъгълник. => Следователно, сборът от вътрешните му ъгли е равен на 180 градуса.
1)всички A имат свойството B
2)X э A
То X има свойството B
1)всички A имат свойството B
2)някои елементи от C имат свойството B
нито един елемент от C не принадлежи на A.
Дедукцията тръгва от предпоставката, че нищо частно не съществува извън общото. Правилно приложеният в разсъждението дедуктив метод осигурява постигане на достоверно заключение, доколкото достоверни са изходните предпоставки за него. Изходните съждения отразяват общи характери на множе предмети, като от тях се извежда съждение за характерист на даден предмет, принадлежащ към това множество. Затова често дедукция бива определ като „слизане“ от общото към частното. Съдържа на формалното разсъжде за дедукцията, може да бъде откроено чрез матрицата на знанието и описано в следното определение за дедукция (респ. дедуктивен метод): Дедукцията извежда единично съждение за връзка на даден предмет с определено множест предмети (P) чрез последователно свързване по необходи (Δ) на общо съждение за характеристики на множеството предмети (U) с частно съждение за характеристики на същото множество, съвпадащи с характерист на предмета. Специфика на дедукцията и дедуктивния метод позволява да се каже, че те осигуряват получаван на толкова достоверен извод, колкото достовер са изходн предпоставки на дедуктивното разсъж.

ИНДУКЦИЯ – от частното към общото разсъждение. И представ логически процес, при който се правят обобщения на базата на набор от наблюдения или друг вид данни. Това е процес на откриване и учене, който започва, когато успеем да видим взаимовръзките между определени феномени и на тази база се опитаме за изведен някаква общовалидна хипотеза. Тоест, ние формираме общи твърдения, чрез наблюдение на огранич брой обекти или събития. Например, ако всички наблюдавани през живота ни здрави котки са с четири лапи, бихме могли да заключим, че всички котки имат четири лапи.
1)A1 всички притежават свойството B
2)A2 всички притежават свойството B
3)A3 всички притежават свойството B
$A \equiv A1 \cup A2 \cup A3$ пълна индукция (A притежава свойстовото B)
$A^* \geq A1 \cup A2 \cup A3$ непълна индукция (A* притежава свойство B, но сме сигурни в %)
индукцията е „противоположност“ на дедуктивното разсъжде и метод, за нея може да бъде формулирано следното определение: Индукция извежда общо съждение за връзка на дадено множество предмети с определен предмет (P) чрез последователно свързва по вероятност (Δ) на частно или единично съждение за характерист на предмета (U) с частно съждение за съвпадащи с тях характеристики на част от множеството предмети (x). В този смисъл могат да бъдат и обобщени препоръките при използването на индуктивния метод с оглед постигане на по-висока степен на правдоподобност:
1. Пълното прилагане на индуктивния метод (т. нар. пълна индукция) е възможно и води до получаване на достоверно знание тогава и само тогава, когато се отнася да крайно множество предмети.
2. Степента на правдоподобност на обикновената (респ. непълната) индукция) може да бъде повишена с увеличаване на броя на изходните предпоставк.
3. Разкриването на по-дълбока съществена връзка между характер и предметите осигурява максимално възможна правдоподобност на индуктивното заключ.
4. Индуктивното разсъждение може да бъде извършено въз основа на: разкриване на причинно-следствени връзки; разкриване на необходими съществени характеристики на предметите; просто преброяване на случаите, отразени в съжденията; изпозване на статистически методи и установяване на честотата на поява на дадени характеристики или събития. Индукция може да бъде пълна – когато можем да наблюдаваме всички елементи от даден вид или непълна

АНАЛОГИЯТА – може да бъде изразена в нейната най-опростена форма с логическата схема: ако A и B притежават характерис a, b и c, и ако A притежава и характе d, то следва, че и B вероятно притежава характеристика d.
Аналогията е опосредствано разсъжде на равнището на частни съждения, при което се прави „извод за принадлежността на определен признак на изследван единичен обект (предмет, събитие, отношение или клас) на основата на сходството на негови съществени характеристики с друг вече известен единичен обект“. Аналогичният метод търси сходство чрез последователното прилагане на закона за транзитивността.
Прилагането на закона за транзитивността поставя аналогията в група логически методи, основани на същия закон и разглеждани като нейни разновидности, наречени традуктивни методи. Някои разглежд аналогията като индуктивен метод въз основа на сходството между аналогия и индукция при извеждането на вероятностно заключени

ЕВРИСТИКА – подход за решаване на сложни задачи като се изработят всички възможни варианти, защото няма как рационален вариант за бързо намиртане на решение. Подход за решаване на сложни задачи с много варианти, при които въз основа на разумни критерии

проверката се ограничава до малък брой от възможните варианти на решението.
Кога се използува евристиката:
а) задачи, за които не са известни алгоритми
б) известните алгоритми изискват изпробването на голям брой варианти.
Евристиката е рационална идея за бързо намиране на решение. Основава се на опит, интуиция. Пример: при игрите защото се знаят комбинациите с по високи оценки основаващи се на опита.
Свойства:
Дедукция – намаляване на броя на възможностите
Евристика
-Не е алгоритъм и не гарантира винаги решаване на задачите
-Използване на евристика е високо ефективно в сравнение с прости алгоритми.

11. Моделиране на Интелигентна дейност ИД (поведение). Механизъм за у-ние на ИД. ИИ дефинига две основни измерения. Едните са свързани с мисловните процеси и мотиви, докато другите се адресират до поведение. Има 4 подхода: да действаш като човек (машината на Туринг); да мислиш като човек - (познавателен) подход;
; да действаш рационално - подход на рационалния агент; да мислиш рационално - „закони на мисълта - логика“. Едните измерват успеха по отношение на вярност към човешкото поведение, докато другите го измерват срещу идеалната концепция за интелигент която ще наричаме *рационалност*. Основни аспекти на интелигентно поведение:
- Перцепция – възприемане на информация от сензори, интерпретиране и представяне с цел натрупване на знания за средата.
- Действане – представлява формиране на команди и активиране на изпълнителните устройства на интелигент система с цел промяна на средата. Може да бъде и чисто информационен акт – напр. Разсъждение
- Разсъждаване – обработки върху сензорно получени знания и други налични знания с цел решаване на задачите.
- Заключване, обучаване, вземане на реш, планиране, диспетчеризарене.

Основни форми на ръзсъждаване са: Заключение (извеждане на нови хипотези в неявен вид в базата от знания), обучение (представлява обобщаване на предишен опит с цел промяна или извеждане на знания за усъвършенстване на поведението), избор на различни варианти на решения, планиране (формиране на последователност от описания на действия, чието изпълнение води до целта), разпределяне на общи ресурси с цел реализиране на процеси по перцепция, разсъждаване и действие. Една система е рационална, ако прави "правилното нещо", като се има предвид, това, което знае. Да действащ рационално: подход на рационалния агент. Агент това е нещо, което възприема средата чрез рецептори или сензори и въздейства върху нея, чрез ефектори. В най-широк смисъл този термин обхваща хора, роботи и програми.
Рационален агент е този, който действа така, че да постигне най-добър резултат. Агента се третира като черна кутия която има следния модел: *Агента притежава* възможност за перцепция на среадата; база знания; някои от тези знания определята множество от целите на агента; притежава възможност да взаимодейства със средата – да изпл. Множество действия; действията са рационални подчинени на принципа на разционалност.
Изкуств интелкт обединва различни области на науката: философия , психология и , биология, физиология, лингвисти, логика, мате матика (в т. ч. инфоматика (КИ)), и др. Архитектура на интелигентните системи) Наличието и обработката на знания е основният белег отличаващ интелигентити системи от останалите.
Важни характеристики на интелигентните системи:
- Решават неструктурирани задачи
- Обработват знания
- Обучават се – с учител, без учител или подпомагано
- Използват специалните подходи на ИИ за намиране на решения (някой от тях са търсене в пространство от състояния, логически разсъждения, съпоставка по образци и др.)
От 1956 година на конфернцията в Дартмут развитието на ИИ се характеризира с ДВА ПОДХОДА. В основата на *знаков* или *символния подход* е хипотезата на Алан Нюел за физическата символна система, че интелигентн поведение изисква символни разсъждения върху символно представени знания. Знак е буква от азбука, графема, звук, която притежава едновремнно 3 свойства: синтаксис, семантика и прагматика. Синтаксиса определя начина на представяне – форма, цвят, компненти на даден знак. Семантиката неговия смисъл – обочначението чрез знака понятие. Прагматиката – асоцирания с него обект – действия които трябва да се визвършат при възприемането на знака. Представянето на знания е на ниво смислова информация (*знания*), а не на ниво данни. Интелигентните системи в това направление като следствие обработват смислово заредени структури, наречени модели за представяне на знания – *Продукц с-ми*, фрейми, лог програмира.
Експертните системи са приер. Но човеш рефлекс не може да се опише с знания.
Числовият или поведенчески подход е основан на хипотезата за пораждащата се функционалност – според нея ИИ може да се породи от голям набор елементарни поведения без централизир управле над тях. Тук се включват невронни мрежи, генетични алгоритми, симулационни програми и т.н. Този подход моделира подсъзнателн поведение. Тя предпола че интелигент поведение може да се породи от голяма колекция относително независим, елем не преполагащи разумност поведения, без каквото и да било централиризано поведен Невроните – сами по себе си не са интелигентни. Двата подхода се използват в различни ситуации. Те не си противоречат а се допълват. Репродукт (предварително е извест послед от действия и с алгоритм език може да се опише) и интелектни задачи – послед от задачи не е известна и трябва да се генерира в процеса на самото решаване.

12. Предметна област. Знание и факти. Метазнания. Процедурни и декларативни модели. Съпоставка.
Съвокуп от всички такива обекти, за които искаме да изразим някакви сведения, представл обща основа за разбирането и решаването на 1 зад се нар *предметна област*. Предметът е обект подлеожен на разглеждане. Предметът е резултат от абстракция на реалния обект, резултат от огрубяване н фрагмент от действит при което се игнорира част от безкрайното многообразие от св-ва и взимовръзки на обекта. И => за формализ се използват мат. Понятия като множество и подмножество. ИИ е познавателна и инженерна научна област, която се занимава с разбирането и синтезирането на поражданото от интелекта ителелигентн поведение (ИП) – взаимод м/у среда и система, х-рно за което е възможността да се знае, разсъждава и действа целенасочено. Като познавателна наука ИИ се занимава с формиране на терминология, модели и теории с цел разбиране на принципите, които правят ИП възможно т.е. научна цел е разбиране на принципите които правят ИП възможно а като инженерна цел – изграждането на ИС. Ако искаш от предния въпрос пиши за 2та подхода.
ЗНАНИЯ И ФАКТИ.
Знания - Съвкупност от факти от една област, които едно лице е усвоило. В изкуствения интелект за разлика от лингвистична гледна точка под знание се разбира обобщена и формализирана информа за дадена предметна област. В в СИИ границата между тези понятия не е ясно изразена. Приемема се, че данните и знанията са 2 страни на категорията - понятие информация. Данните са допълнителна информа за знанията; те носят информация за обектите; те са това, което наричаме факти в естествения език. Под

факт ще разбираме общоизвестните в дадена предметна област истини или обстоятелства. Знанията от своя страна са общата и неизменна част от информацията; това са законите и взаимовръзките в самата информация.
Например при пресмятане на дължината на конкретна окръжност данни за нас са конкретната дължина на радиуса и стойността на числото. Знанието в този пример е формулата, по която пресмятаме дължината на окръжност. С други думи знанието за нас е взаимовръзката между конкретните данни. Можем да обобщим, че върху данните се осъществяват действия, а знанията се използват за вземане на решения, изводи и съставяне на планове.
ТИПОВЕ/ВИДОВЕ ЗНАНИЯ: Съществуват три основни типа знания в системите с изкуствен интелект:
Знания за обектите в предметната област;
Знания за връзките между обектите и знанията за тях;
Метазнания – знания за самите знания. (пр. за стр-та на едно правило) описват самите знания; как те се групират, в каква структура се подреждат, кога се използват; имат ли отношение към проблема или нямат; колко често се използват. Знания за местоположението на знанията, което ни е необходимо. Те са модели от гледна точка на:
-изграждане на база данни
-постигане на по висока ефективност при изграждане на базата
-обяснение на самите знания. Пример: Правило: Ако има утечка от сярна киселина; То: да се използува вар.
Обяснение (метазнание): Варта неутрализира киселината.
ФОРМИ ЗА ПРЕДСТАВЯ НА ЗНАНИЯТА. – Предств на знания е множеството от синтактични и семантичн конвенции, които правят възможно описанието на обекти, свойства, отношения, процеси, и др. в опред предметна област. Представянето и използв на знания (ПИЗ) е област от изкуствения интелект, целта на която е знанието да се представи символно по такъв начин, който да улесни извеждането на заключения от знанията, с които разполагаме, използвайки по-скоро разсъждения отколкото действия. Съществено за системите с изкуствен интелект е преминаването от работа с данни към работа със знания.
ФОРМИ / модели за представяне на знанията. 2 осн – *декларативни и процедур.* Съществуват два основни типа формализми (подходи) за ПИЗ - формализми от *декларати* тип (декларативни знания) и формализми от *процедурен (алгоритмичен)* тип. При първите съществен е начинът на представяне на
Знанията -«какво знания», докато при вторите съществен е начинът на използване на знанията – «как». При декларатив формализми знанията се представят явно, а при процедурните, знанията са
неявно представени чрез процедурите в програмнат система. Изборът на типа формализъм зависи от конкретната задача.
А.Процедурни-*начина*, по който се строи нещо – изразени под формата на формата на алгоритъм.
Б.Декларативни-описание най-широко разпростране
При декларативните механизми въпросът е Какво представлява? Площ (тип фигура(име(квадрат), (страна(б))),bна2);
Декларативните позволяват знанията да бъдат относително обособени от процедурата за тяхната обработка. => дава възможност да за извършване на манипулации (корегира и допълване по дедуктивниен път) по същество със самите знания, което означава че знанията може да се ползват даже и за таквиа цели които не са известни предварително в нач момент. Процедурите за извод (алгоритмите на интерп татора) при декларативните
представ са достатъчно общи и могат да бъдат използвани в различни предмтни области. И при тях измененията в БЗ се правият значително по лесно от колкото при процедурните представ.
Процедурите за извод са достатъчно общи и работят с различни знания. Обобщенията на знанията е много лесно. Знанията и връзките между тях са разделят. Те имат по голяма ефективност (по време и памет).
При процедурните основен въпрос е Как се прави? Нужно е знание + начините за неговото използване знание и връзки между тях са в тясно единство свързани. За да бъде задействан алгоритъмът (поредица от стъпки), данните трябва да бъдат подходящо представени – манифестирани в само при изпълнение на указаните команди в стриктна последовател. Високо ефективни от гледна точка на времето. Бързо се работи с тях. Но зависят от съставителя им – да съответстват на конкретната ситуация. *Процедурите* широко се използват в демонни с-ми. При тях е характерно, че данните се включват в самата процедура. Връзките между процедурите се осъществяват чрез предаване на параметри. Основен проблем е внасянето на промени в базата данни. Който ги използва трябва да планира щателно всички ситуации и да има начин за изход от непланираните ситуаци. Тези знания трудно може да се приложат за решаване на проблеми различни от първоначал за който са създадени. Такива занния се използват доколкото са осмислени от съставителя на алгоритъма. Търси съчетаване на 2та подхода.
Модели за изграждане на процедурни схеми:
-описват се основните понятия в предметната област чрез разпознав алгоритми;
-с тези понятия , следвайки правилата на композицията , съобразявайки се с ролята на предметната област правим нови понятия(производни).
Всяко понятие се представя чрез съдържание и логическа форма.
Основно предимство на процедурния метод е висока ефективност, бързо действие, ъправлението се прави директно. Недостатък-в тези системи е почти невъзможнода се изгради с-ма за обяснение.
Декларативните методи използват различни възможности за описание на знанията.
Мрежови модели (семантични мрежи). Модели с причинно-следствен връзки:
If условие, then действие
-продукционни модели
-фрймови модели
-логически (процедури) модели.

13. Представяне и използван на знания (ПИЗ) чрез процедури – същност и характерна особености. Предимства и недостатъци на процедуранит модели.

Виж 12ти въпрос накрая: ФОРМИ ЗА ПРЕДСТАВЯ НА ЗНАНИЯТА. ;)
Много от човешките знания по същество имат процедурен характер (всички алгоритмични знания). Голяма често от метазнанията на СИИ също имат процедурниен х-р. С помощта на процедури могат да се описват предметни области. За компютъра най-лесно е с последователност. Всеки алгоритъм има две страни:
-съдържание- онези блокчета, от които е изграден; -логическа форма - кога отговърът е “Да”. Основни концепции в ИИ. Процедурният тип ПИЗ е сравнително по-
късно създаден модел, възникнал след основните декларативни схеми - ПИЗ чрез средствата на предикатната логика от първи ред и чрез семантични мрежи. Тук
става дума за обособяване на процедурите като модел за ПИЗ, тъй като всяка програма “носи” в себе си своите знания и в този смисъл те са представени
чрез нея. Когато се говори за процедурно ПИЗ, се имат предвид не стандартните ПС, а системи от програми, наричани още демони (Уинстън, 1977) - процедури, в които са закодирани знанията за действията в определени характерни ситуации и които се активират само при настъпване на дадена ситуация. За разлика от стандартните ПС, за които е характерна йерархична структура, за базите от процедури (демони) е типична хетерархична структура. За разлика от декларативните типове, където основната тежест пада върху ПИЗ, при представянето чрез процедури е съществен начинът на използване на

знанията (Бар, Файгенбаум, 1981).

ОБЩА ХАРАКТЕРИСТ:

Когато се говори за процедурно ПИЗ, се имат предвид не стандартните ПС (простр на състоянията), а системи от програми, наричани още демони - процедури, в които са закодирани знанията за действията в определени характерни ситуации и които се активират само при настъпване на дадена ситуация. За разлика от стандартните ПС, за които е характерна йерархична структура, за базите от процедури (демони) е типична хетерархична структура. Два са начините за представяне на знания:

а. чрез *релация*, която в естествения език (ЕЕ) се изобразява чрез предикат, например Иван е син на Петър или по-формално - е син на (Иван, Петър);

b. чрез *алгоритъм* - знания от тип know-how (знам как), които са кодирани или на ЕЕ (ест език), или на ЕП (езика на операндите). И в двата случая се използват два езика: алгоритмичен език (дори да е част от ЕЕ) и език на операндите.

Стандартните ПС се изобразяват в общия случай върху граф, т. е. структурата им е хетерархична и много по-рядко - йерархична, която се изобразява върху дърво. Най- сетне и модулите на стандартните ПС се активират в общия случай само при настъпване на дадена ситуация например при получаване на определени междинни резултати. Демоните са процедури, които се използват в ИИ. Те обикновено са от изчислителен характер например пресмятат релации.

ПРЕДСТАВЯНЕ НА ЗНАНИЯТА:

Семантичният -смисловите елементи при процедурни формализъм са самите процедури. Например знанието за дължината на окръжността с радиус r се изразява чрез процедурата procedure Cr (r, C: real);
begin read (r); if r<O then error; C := 6.283185*r;
write (C) end;

СТРУКТУРА НА БЗ:

При създаването на една процедура в нея трябва да се вложат знанията за конкретната ситуация и да се създадат необходимите връзки с останалите процедури от БЗ. За разлика от декларативнит представяния, където знанията и връзките между тях са разделени, тук както самите знания, така и връзките са заложени в процедурите. Знанията се изразяват чрез семантиката на процедурите, а връзките между тях - чрез връзки между отделните процедури, т. е. чрез обръщения към други процедури с предаване на параметри. (процедурен език примерно Haskel).

ИЗПОЛЗВАНЕ НА ЗНАНИЯТА:

Характерни за процедурните типове ПИЗ са ефективността и бързината при използван на знанията. Тъй като връзките са заложени като обръщения в самите процедури, използването се извършва само чрез подаване на данните за конкретната ситуация. Управлението на процеса се осъществява от самите процедури въз основа на тяхната вътрешна логика, но по тази причина не могат да се извършват непланирани действия и да се отговаря на непредвидени въпроси.

ДОБАВЯНЕ И ИЗМЕНЕНИЕ В БЗ:

Изменението на знанията е основният проблем при процедурното ПИЗ. Трудностите идват от това, че връзките между знанията са твърдо заложени в процедурите. Изменението само на една процедура може да доведе след себе си до необходимост от измене на голяма част от процедурите на с-мата (например при добавяне на допълнителен параметър в някоя процедура). Добавянето на нови процедури от своя страна не се изчерпва с механичното им долепяне към БЗ, а води до изменението на редица от съществува процедури, за да се създадат необходимите връзки (добавяне на обръщения към новата процедура, предаване на параметри към и от нея и др.). От друга страна, проблемът за добавяне в БЗ е относително по-прост, отколкото проблемът за промяна в структурата на стандартни ПС.

ХАРАКТЕРНИ ОСОБЕНОСТИ И ПРИЛОЖЕНИЯ:

Основно преимущество на процедурните ПС са техните относително по-високи ефективност и бързодействие, което се дължи на факта, че управлението се предава директно от процедура на процедура в зависимост от текущите данни. Директните връзки са заложени в самите процедури (връзките са обръщения към други процедури), което води до директен извод.

Освен особеностите на всички процедурни схеми за ПИЗ това представяне на знания чрез процедури има и редица особености, характерни само за този тип схеми. Така например процедурното представяне предоставя редица удобства при моделиране на процеси, тъй като дадена ситуация се обработва от конкретна процедура и управлението се предава директно точно на тази процедура, която притежава знания за новата ситуация (при логически системи всеки път се претърсва цялата БЗ). Процедурните ПС са много удобни и при представяне на евристични знания. От друга страна, връзките са относително принудителни, което води до липса на модулност в системата и до трудности в управлението на процеса отвън. Освен това във всеки момент управлен се намира в дадена процедура, което води до липса на общ поглед върху състоянието на нещата. Друг недостатък на процедурните ПС (прод система) за ПИЗ е, че трудно се реализират системи за обяснение на резултата. Едно от основните преимуществ на процедурните ПС е това, че те са ориентирани към решавания проблем, което също е причина за тяхната относително висока ефективност спрямо декларативните модели за ПИЗ. Процедурните ПС за ПИЗ имат голямо приложение при управление на роботи, управление на процеси и др., където преобладават алгоритмичните знания. При декларативния модел базата знания и обработката са отделно. Те са подходящи за описателни знания с много логически съждения. Дели се на модулно и мрежово, като при модулните модели се използват логически и продукционни правила. При мрежовите модели най-ярък представител са семантичните мрежи, където в явен вид се задават взаимоотношенията в явен вид. Процедурните модели изграждат заедно базата знания и оперативната част. Процедурно-декларативните съчетават предимствата на двата вида. Основни представители са фрейми и сценарии. В реалните задачи има предпочитание за използване и на трите групи модели. Така, че отношението между тях се решава за всеки конкретен случай.

14. Съждително смятане (СС). Осн закон на СС. Нормални форми (НФ) в СС - преобразуване на НФ.

Най-простия *логически модел* е *съждително смятане* - Propositional calculus или Logic (PL) - всяко твърдение се разглежда от гледна точка на своята истинот или неистиност.

Съждителното смятане е сравнително прост декларативен метод за представяне на знания.

Всяко съждение е атом и се разглежда като неделимо цяло.

СИНТАКСИС граматика (Език на съждителното смятане) Основни понятия в съждителната логика са:

Атомът или отрицанието на атома се нарича литерал. Например простото изречение “Днес е слънчев ден” е съждение. За всеки конкретен ден то или е истина (true), или не е истина (false). Нека отбележим това твърдение с G. Тогва за слънчев ден се записва G= true, а за ден, в който не грее слънце - G= false. За отрицани на G записът е ¬ G.

Ограничеността на съждителното смятане - За да представим ден, в който до обяд е гряло слънце, а след обяд е било облачно, трябва да излезем от ограничението надвоични величини и Булевата алгебра, като използуваме многозначна или размита алгебра.

 Твърдения (Съждения) - бележат се с латински букви, например P и изразяват твърдения, например: „навън вали“. Съжденията могат да са верни (да приемат ст-ст Истина) или да са неверни (да приемат стойност Лъжа); *Съждение* за което се твърди че е истина се нар *твърдение*.

□ Съждителни връзки (литералите са свързани чрез отношения):
„∧“ (логическо и) конюнкция),,
„∨“ (логическо или) (дизюнкция),
„¬“ (отрицание),
„→“ / ⇒ (импликация) - при истинност на предпоставка следва истинност на следствието (ако… то…),
„↔“ / ≡ (еквивалентност или равнозначност)
{¬,∧, ∨,⇒, ≡} – множ от лог. съюзи.
□ Скоби „( )“ е множеств от логически съюзи вътре. Съжденията са елементарн (атоми) и съставни. От ел може да се образуват съставни чрез преобраз посредством частицата «не» или чрез съчетаване на с помощта на съюзите и «тогава и само тогава», „↔“ и «ако и само ако» „→“.
От свързани със знаци за логически операции атоми в съждителното смятане се изграждат правилно построени формули (ППФ).
Правилно построена формула (ППФ): ППФ се строят по следния начин:
1).Атомите са ППФ
2). Ако G е ППФ, то ┐G също е ППФ
3). Ако G и H са ППФ, то G или H, G и H, G→H, G≡H са ППФ
4).Няма други ППФ,освен правилата от 1). до 3).
Представянето на знания чрез съждителна логика се осъществява чрез правилно построени формули (ППФ). Под ППФ разбираме последователно от твърдения, свързани с допустими съждителни връзки от съждителната логика. Може да се представят като дървориден граф състоящи се от единичен корен – атом. Тя е низ от атоми, свързани със знаковете за логически операции от съждителното смятане и кванторите за съществуване (3) и всеобщност (всички), като кванторите могат да бъдат използвани само за (пред) променливи (но не и за (пред) функции - затова е смятане от първи ред). Всяка елементарна ППФ има своя стойност.
А) ¬ G – логическо отрицание
б) (G ∨ H) – *дизюнкция*;
в) (G ∧ H) – *конюнкция*;
г) (G →H) – импликация от G към H (следва);
д) (G ↔H) – логическа еквивалентност (if and only if – iff) или еднозначност ≡;
СЕМАНТИКА на съж. Смятане – Атом (използв символ вярно true- “⊤, невярно-false “⊥”), литерал, Формула (F)
Всяко присвояване на конкретни стойности (в таблица на истинност) на атомите във формула се нарича нейна интерпретация. Ако една формула съдържа n на бр. атома => тя ще има n бр интерпретации и с нея 2 на n асоц с нея различни предметни области (таблица на истинност). Всяка формула (в случая X) от СС може да се класиф. като *общовалидна* – (тя е истина при всички възможни нейни интерпретации), *противоричива* – за всички е F и *зависима* / удовлетворима когато съществува поне 1 интерпретациия за която X е лъжа и поне 1 за която X е истина;
Да се оцени коректоноста на ф-ма в някаква предмаетна област означава да се опр инстинноста на ст-ста на формулата X в зависимост от нейните атоми.Множеството на всички такива наредени двойки (X, I) че I удовлетв. X се нарича реалация удовлетворимост.
Записът “|=i” – означава “формулата X се удовлетворява в i” и “|=/=I” не се удовлетвирява. Всяка интерпретация която удовлетворява X се нарича *модел* на X.
ОСН ЗАКОНИ – (асоциативни).
Логическа еквивалентно на формулите е релация на две формули такава , че формулите A и B са еквивалентни (A≡B) за всЯка оценка v имаме v(A)=v(B).
Еквивалентността (≡) е логически закон/и:
1. A ∨ B ≡ B ∨ A - *комутативни закони* и инвертирано A ∧ B ≡B ∧ A
2. (A ∨ B) ∨ C ≡A ∨ (B ∨ C) - *асоциативни закони* и инвертирано
3. A ∨ (B ∧ C) = (A ∨ B) ∧ (A ∨ C) *дистрибутивни* закони и инвертирано - A ∧ (B ∨ C) = (A ∧ B) ∨ (A ∧ C);
4. A → B ≡¬ A ∨ B
5. A ↔B ≡ (A → B) ∧ (B → A);
A ↔B ≡ (A ∧ B) ∨ (¬A ∧ ¬ B);
A ↔B ≡ (A ∧ ¬B) ∧ (¬ A ∨ ¬ B);
6. A ∨ □ ≡ A; A ∨ ■ ≡ ■;
7. A ∧ □ ≡ □; A ∧ ■ = A;
8. A ∧ A ≡A; A ∨ A = A – зак *едемпотентност* ;
9. A ∨ ¬ A ≡ ■;
A ∧ ¬ A ≡ □ закон за *допълнителност*
10. ¬ (¬ A) ≡ A ; закон за *двойното отрицание - инволюция*
11. ¬ (A ∨ B) ≡ (¬ A) ∧ (¬B);
¬ (A ∧ B) ≡ (¬ A) ∨ (¬ B) закони за отрицанието (закони на *де Морган*)
НОРМАЛНИ ФОРМИ (НФ).
Летерал е атом или отрицание на атом. Литерал съдържащ (не съдържащ) отрицание се нар. отрицателен (положителен) литерал. *Клауза е дизюкция от кр бр литерали* (m1 ∨ m2 ∨ m3 ∨ … ∨ mk) където за всяко mi е литерал. Клауза която не съдържа нито 1 литерал се нарича *празна* клауза □. Празната клауза винаги е лъжа защото тя изобщо не съдържа литерали които са предмет на евентуално удовл на някаква интерпретация.
Най често се използват 2 нормални форми:
Нека m i е конюнкция от литерали. Тогава формул m1 ∨ m2 ∨ m3 ∨ … ∨ mk за i>= 1 а mi е куб (конюкции от атоми) се нарича *Дизюнктивна нормална форма (клауза)* (ДНФ). Нека Mi е дизюнкция от литерали. Тогава формулата M1 ∧ M2 ∧ M3 ∧ … ∧ Mp където i>=1 и всяко Mi е клауза (дизюнкции от атоми/литерали) се нарича *Конюнктивна нормална форма* (КНФ).
Произволна формула може да бъде преобразувана в ДНФ или КНФ чрез използване на логическите закони по долу. *Законите на съждително смятане* се използват като ПРАВИЛА ЗА ТЪЖДЕСТВЕНО ПРЕОБРАЗУВАНЕ на ППФ в НФ.
Преобразуване на формули в нормална форма (НФ):
1. Премахват се ≡ и → (закони 5 и 4).
2. Отстраняване на отрицанията на над повече от един атом, т.е. изнасяне на ¬ непосредствено пред атом. (закони 11).
3. Преобразуване чрез използване на останалит закони. Ако формулата (M1 ∧ M2 ∧ M3 ∧ … ∧ Mk) → G е общозначима, то G е логическо следствие на M1, M2 , M3, …, Mk.
G е логическо следствие на M1, M2 , M3 , …, Mp , тогава и само тогава, когато ■ е логическо следствие на M1, M2 , M3 , …, Mp , G
Ако формулата (M1 ∧ M2 ∧ M3 ∧ … ∧ M k → ¬ G) е противореч, то G е логическо следствие на M1, M2 , M3 , …, M k .
G е логическо следствие на M1 , M2 , M3 , … , M k тогава и само тогава, когато □ е логическо следствие на M1, M2 , M3 , …, M k , ¬ G. Пример:

15. Принцип на резолюция в съждителното смятане

Виж pdf в моб. ;) ако можеш де…
Метод, който оперира върху съвокупност от твърдения въз основа на които могат да се изведат др. твърдения. Използва се за да се доказват теореми. В тях се използва отрицанието на твърдението което ще се доказва. Предимсто на работа с отрицанитео вместо с самото твърдение се състови в това че появата на противоречие в доказателството означава че търсения резултат е доказан! Обикновено начина на изразяване на дадено тръдение е формлирането му като последователност от елементарни изрази - наречение *литерали* всеки от които е функция на 1 или повече аргумента. Литаралите са разделение с OR (и се изпуска често).
1 от причините за широката приложимост на метода на резолюц е че той дава възможност да се въведат толкова различни видове атомарни ф-ми (атоми) колкото е необходимо за формулировката.
Атомарната ф-ма прилича на алгебрична функция и представлява утвърждение което е истина или лъжа. Ако една атомарна ф-ма означим с P то тя изобразява реалцията “е част от”. Формулата P (x,y) е истина ако x е част от y, и лъжа в всички останали случаи. Нека пръст (f) е част от длан (h), длан (a) е от ръка а ръка е от човеч тяло (m). да се док. че пръста също е част от чов тяло. Или може да запишем формулата ако P(x,y), и P(y,z) то => P (x,z). Форм се записва като съвокоупност от литерали но Вместо да се докаже че АКО първите две са истина => и 3тото е истна се казва че или 3тото е

истина или 1 от първите 2 са лъжа. И добавяме 5 твърдения:

1. P(x,z) ¬P(x,y) ¬ P(y,z).
2. P(f,h)
3. P (h,a)
4. P(a,m)
5. ¬P (f,m)

Основната операция от която зависи метода е: *Резолюцията* на 1 ритерал с друг. Т.е трябва да се намерят 2 литерала така че чрез допустими замествания 1 да се превърне в отрицание на другия. Или заместваме 2рия лит. от 1 с 2ро трърдение. Двата литерала са P(f,h) и ¬P(x,y). Те могат да се доведат до вид при който 1 да отрича другия ако в цялото твърдение 1 x се замести с f, y – h. тогава твърдение 1 добвива вида P(f,z) ¬ P(f,h) ¬ P(h,z) а втърдение 2 остава непроменено. Тъй като съгласно твърдение 2 P(f,h) е истина то 2рия литерал от пременето твърдение 1 омда се се елеиминира и се получава 6. P(f,z) ¬ P(h,z). С понаташни резолюции се довечждане на твърдение несъдържащо никаква литерали което показва противоречие => теоремата е доказана. Резолюцията на литерал от m-елементрно твърдение с лирарал от n-елементно води до получаване на твърдение с брой на лиералите не повече от (m+n-2).

Нека са дадени две дизюнктивни нормални формули (ДНФ):

M1 = a ∨ b;

M2 = ⎤ a ∨ с където b и с са дизюнкции от литерали. Тогава

M1^ M2 = M1^ M2 ^ (b ∨ c);

(b ∨ c) се нарича *резолвента на* M1 и M2.

ДЕФИНИЦИЯ: За произволни две дизюнкции c1 и c2 от литерали (т. е. атоми или отрицания на атоми), ако съществува литерал в c1, който е противоположен (т. е. еквивалентен на отрицанието) на някакъв литерал от c2, то ако задраскаме тези елементи, съответно от c1 и c2, и построим дизюнкцията на останалите елементи на c1 и c2, получената формула се нарича резолвента на c1 и c2. Дизюнкцията от литерали се нарича клауза (дизюнкт). Резолвентата на две клаузи - c1 и c2 е следствие от тези клаузи. Правилото за образуване на резолвенти се нарича *правило на резолюцията* (ПР).

СВОЙСТВО НА РЕЗОЛВЕНТАТА – Нека са дадени 2 клаузи C1 и C2. Тогава коя и да е тяхна резолвента R е логическо следствие на C1 и C2. Резолвентен извод на клаузата C от S е такава кр последователност от C1, C2… Ck от клаузи че: C=Ck, всяка Ci или пренадлежи на S, или е резолвента на предходно изведени по ПР клаузи. Извод на празна клауза от S се нар *опровержение*. Изводът с използването на резолвенти се изпълнява в следната последователност:

1.Избират се две клаузи, които съдържат един и същ литерал в различна форма.

2.Получената резолвента се добавя към множеството клаузи.

3.Действува се докато се получи празна резолвента т.е. □

Ако твърденията от системата S не могат да се представят чрез клаузи (дизюнкции от литерали), тогава привеждаме тези твърдения (ППФ) в КНФ и в системата включваме отделните клаузи, които участват в съответните КНФ.

16. Предикатно смятане от първи ред. First-Order Logic (FOL). Осн понятия. ППФ, НФ, преобраз в НФ. Закони за предик смятане от първи ред. First-Order Logic (FOL).

Само малка част от изреченията в естествения език могат да се представят със съждителна логика. Предикатното смтане от първи ред е формален език, чито основна цел е да даде възможност за символно представне на логически доказателства в математиката. С негова помощ може да се опише голмо разнообразие от твърдени. Създадел – немски мат. Готлоб Фреге. Логиката от първи ред е по-изразителна от съждителната. Тя е формална логическа система - обобщение на съждителното смтане. Синтаксисът на логиката от първи ред е разширение („надгражда“) този на съждителната логика, като въвежда още три понятия: *термове, предикати и квантори*. - Въвеждат се предикати, правила и квантори и се използва по-богат набор от термини

- Въвеждат се променливи – чрез тях косвено се представя група обекти с еднакви характеристики или представянето на знания в предикатната логика е чрез терми за представяне на фактите за обектите и правила за изразяване на отношенията.

variables x,y,z,⋯

constants a, b, c, ⋯

functions f,g,h,⋯

Основно при това представяне е, че задачата се задава като съвкупност от логически формули и нейното решаване се свежда до решаване на логически извод. Той показва, че целта е логическо следствие на зададените формули. За логическия извод са необходими система от аксиоми и теореми, които заедно с метода на рез. Използват се променливи. Чрез променливите косвено се представя група обекти с

еднакви характеристики. Променливите са за обозначаване само на

обекти, а не на отношенията между тях. Представянето на знания (синтаксиса) в предикатната логика е чрез:

- Терми за представяне на фактите за обектите;
- Правила, предикати и
- Квантори за изразяване на отношения и свойства

Понятието TERM се определ по следния начин:

1. Константите са термове;
2. Променливите са термове;
3. Ако f е n-местен функционален символ, а t1, t2,…, tn са термове, то f(t1, t2,…, tn) също е терм;
4. Няма други термове освен построените чрез правилата 1-3.

ПРЕДИКАТ – зависимост между една или повече променливи, която приема само две стойности – true u false. Средство за представяне на релации от дадена предметна област.

Ако променливата X приема стойност от нашето родословно дърво, то предикатът е истина. P(x) = T – едноместен предикат; P(x,y) – двуместен и n - местен p (x1, x2..xn) предикат представящ релацията P.

КВАНТОР – съществуват два квантора

(Vx) P(x) - всеобщност

Vx – кванторен комплекс

P(x) – предикат

За всички x P(x) е истина

□ x Q(x) съществува x, за което Q(x) е истина.

Оценка на предикатната логика: Представлява пълен и достоверен модел на разсъждения; По естествен начин се описват обектите и отношенията, като отдясно задаваме предпоставките и от ляво е заключението.

Съществуват следните квантори: квантор (quantifiers) за *всеобщност* (∀), ∀x. F[x]– “за всички x, F[x]” и квантор за *съществуване* (∃), ∃ x . F [x] - “съществ. X. такова че F[x].”

Чрез кванторите за всеобщност и съществуване можем да изразяваме, че дадено твърдение е вярно за всички обекти (твърдението е универсално) или че е вярно поне за 1 обект (твърдението е екзистенциално).

СЕМАНТИКАТА ПРИ ЛОГИКАТА ОТ ПЪРВИ РЕД се изразява отново чрез правила за извод. Примерно: „Всеки понеделник и всяка сряда отивам при Джон за вечеря.“ На езика на предикатната логика от първи ред то би изглеждало по следния начин ∀ X ((ден_от_седмицата(X,понеделник) ∨ ден_от_седмицата(X,сряда)) → (отива_при(аз, Джон) ∧ яде(аз, вечеря))) или

∀x. p(f(x),x) → (∃y. p(f(g(x,y)),g(x,y))) ∧ q(x,f(x))

The scope of ∀x is F. The scope of ∃y is G. The formula reads:

∀n. integer(n) ∧ n > 2 → ∀x,y,z.

“for all x,

if p(f(x),x)

then there exists a y such that p(f(g(x,y)),g(x,y)) and q(x,f(x))”

това по горе е ПРАВИЛНО ПОСТРОЕНА ФОРМУЛА (ППФ). Тя е низ от атоми, свързани със знаковете за логически операции от съждителното смтане и кванторите за съществиване (∃) и всеобщност (∀), като кванторите могат да бъдат използвани само за (пред) променливи (но не и за (пред) функции - затова е смтане от първи ред).

СВОБОДНА ПРОМЕНЛИВА. Съдържа се във формулата, без да се

включва в областта на действие на никой от кванторите.
СВЪРЗАНА ПРОМЕНЛИВА. Включва се в областта на действие на (∃) или (∀).
Понятието ППФ в предикатното смятане от първи ред се определя рекурсивно по следния начин:
1).Атомите са ППФ
2). Ако F и G са ППФ, то ┐G , ┐F, F → G, F≡G също е ППФ
3). Ако F е ППФ и x е свободна променлива в F, то (∀x) F(x) и (∃x) F(x) също са ППФ.
4).Няма други ППФ,освен построените правилата от 1). до 3).
В сила са всички закони на съждителното смятане, а също и следните закони, които се отнасят до кванторите:
1. (qx) F[x]□G = (qx) (F[x]□G).
2. (qx) F[x]□G = (qx) (F[x]□G).
3. ¬((∀x) F[x]) = (∃x) (¬F[x]).
4. ¬((∃x) F[x]) = (∀x) (¬F[x]).
5. (∀x) F[x]□(∀x) G[x] = (∀x) (F[x]□G[x]).
6. (∃x) F[x]□(∃x) G[x] = (∃x) (F[x]□G[x]).
7. (q1 x) F[x]□(q2 x) G[x] = (q1 x) (q2 y) (F[x]□G[y]).
8. (q1 x) F[x]□(q2 x) G[x] = (q1 x) (q2 y) (F[x]□G[y]).
ПРАВИЛА ЗА ТЪЖДЕСТВЕНО ПРЕОБРАЗУВАНЕ на ППФ в НФ. Преобразуване на формули в нормална форма (НФ):
1. Премахват се ≡ и → (закони 5 и 4).
2. Пренасяне на знака ¬ непосредствено пред атомите във формулата.
3. Преименуване на част от свързаните променли, ако това е необходимо.
4. Изнасяне на кванторите отпред.
НЕДОСТАТЪЦИ:
Монотонна е, получените изводи се считат за верни през целия процес на логическия извод. Те не могат да се отрекат или променят от следващи получени резултати;
Не може да се работи с несигурни знания.
Така че, счита се предикатната логика за подходяща за малки модели, работещи в експериментални условия. Предимството е високата скорост, но не е удобен за хората.

17. Унификация на предикатното смятане от първи ред. Метод на резолюцията.

Видяхме, че формата за прилагане на метода на резолюцията върху формулите от съждително смятане е тяхната КНФ. Аналогично, формата за прилагане на метода на резолюцията върху формулите от предикатното смятане от първи ред е тяхната т. нар. стандартна (скулемова) нормална форма (НФ).
Общият вид на стандартната (скулемовата) НФ е:
F = (∀x1) (∀x2)... (∀xn) (M), където M е формула без квантори, приведена в КНФ. Следователно привеждането в скулемова НФ изисква премахване на ∃ от префикса. Премах. на кванторите за съществуван от префикса. Да допуснем, че (Ør Xr), 1<= r<= n, е най-левият квантор за съществуване в префикса на F (т. е. Ør = ∃ и Øi = ∀, 1 <= i < r, ако r > 1).
Тогава алгоритъмът за премахване на Ør (а след това - и на всички останали ∃ в префикса от ляво на дясно) изглежда по следния начин:
1. Ако в префикса няма ∀ наляво от Ør (т. е. ако r = 1), то избираме нова константа C, различна от всички константи от M, и заменяме всички срещания на Xr в M с C, като премахваме от префикса (Ør Xr).
2. Ако r > 1, т. е. (Ø1 x1), (Ø2 x2),..., (Ør-1 xr-1) са ∀ в префикса преди (Ør xr), то избираме нов функционален символ f, различен от всички функционални символи, които се съдържат в M, заменяме всички срещания на xr в M с f(x1, x2, ..., xr-1) и премахваме от префикса (Ør xr).
Използваните константи и функции се наричат съответно скулемови константи и функции (константи и функции на Скулем), а процесът на премахване на кванторите за съществуване се нарича *скулемизация на формулата F*.
След премахването на кванторите за съществуване от префикса и привеждането на формулата M в КНФ формулата F е в стандартна (скулемова) НФ и е подходяща за прилагане на метода на резолюцията.
Забележка. В системата от клаузи (дизюнкти) участват отделните клаузи, от които се състои КНФ на формулата M. Кванторите за всеобщност не се включват, но се имат предвид. За да може методът на резолюцията да бъде приложен върху дадено множество от формули, остава само един съществен проблем - унификацията (определянето на противоположните елементи в клаузите с цел намирането на съответните резолвенти). При съждителното смятане нещата са прости, тъй като там няма променливи и функции. Тук наличието на променливи и функции прави задачата значително по-сложна. За решаването є са създадени множество формални методи, които няма да бъдат разглеждани тук, тъй като изучаването им е отделна тежка задача. При решаване на конкретни задачи ще извършваме унификацията по интуиция чрез извършван на подходящи субституции, отчитайки наличието на квантори за всеобщност за съответните променливи.

18. Методи за представяне на знания – видове и съпоставка. Виж. 12 въпрос!!!

ФОРМИ ЗА ПРЕДСТАВЯ НА ЗНАНИЯТА.
ФОРМИ / модели за представяне на знанията. 2 осн – *декларативни и процедур.* Съществуват два основни типа формализми (подходи) за ПИЗ - формализми от *декларати* тип (декларативни знания) и формализми от *процедурен (алгоритмичен)* тип. При първите съществен е начинът на представяне на Знанията -«какво знания», докато при вторите съществен е начинът на използване на знанията – «как». При декларатив формализми знанията се представят явно, а при процедурните, знанията са неявно представени чрез процедурите в програмнат система.
Декларативните позволяват знанията да бъдат относително обособени от процедурата за тяхната обработка. => дава възможност да за извършване на манипулации (корегира и допълване по дедуктивниен път) по същество със самите знания, което означава че знанията може да се ползват даже и за таквиа цели които не са известни предварително в нач момент. Процедурите за извод (алгоритмите на интерп татора) при декларативните представ са достатъчно общи и могат да бъдат използвани в различни предмтни области. И при тях измененията в БЗ се правият значително по лесно от колкото при процедурните представ.
Процедурите за извод са достатъчно общи и работят с различни знания. Обобщенията на знанията е много лесно. Знанията и връзките между тях са разделят. Те имат по голяма ефективност (по време и памет). При процедурни основен въпрос е Как се прави? Нужно е знание + начините за неговото използване знание и връзки между тях са в тясно единство свързани. За да бъде задействан алгоритъмът (поредица от стъпки), данните трябва да бъдат подходящо представени – манифестирани в само при изпълнение на указаните команди в стриктна последовател. Високо ефективни от гледна точка на времето. Бързо се работи с тях. Но зависят от съставителя им – да съответстват на конкретната ситуация. *Процедурите* широко се използват в демонни с-ми. При тях е характерно, че данните се включват в самата процедура. Връзките между процедурите се осъществяват чрез предаване на параметри. Основен проблем е внасянето на промени в базата данни. Който ги използва трябва да планира щателно всички ситуации и да има начин за изход от непланираните ситуаци. Тези знания трудно може да се приложат за решаване на проблеми различни от първоначал за който са създадени. Такива занния се използват доколкото са осмислени от съставителя

на алгоритъма. Търси съчетаване на 2та подхода.
Основно предимство на процедурния метод е висока ефективност, бързо действие, ъправлението се прави директно. Недостатък-в тези системи е почти невъзможнода се изгради с-ма за обяснение.
Декларативните методи използват различни възможности за описание на знанията.
При *декларативния модел* базата знания и обработката са отделно. Те са подходящи за описателни знания с много логически съждения. Дели се на *модулно и мрежово, като при модулните модели се използват логически и продукционни правила.*
При мрежовите модели най-ярък представител са семантичните мрежи, където в явен вид се задават взаимоотношения в явен вид. Процедурните модели изграждат заедно базата знания и оперативната част. Процедурно-декларативни съчетават предимствата на двата вида. Основни представители са фрейми и сценарии. В реалните задачи има предпочитание за използване и на трите групи модели. Така, че отношението между тях се решава за всеки конкретен случай. Могат да бъдат посочени следните основни начини за описване на *декларативните* знанията от дадена предметна област:
А. МРЕЖОВИ МОДЕЛИ ИЛИ СЕМАНТИЧНИ МРЕЖИ –
Семантичните мрежи са от декларативния тип формализми за ПИЗ. Освен това те принадлежат към т.нар. структурни формализми, които се характеризирт с това, че се основават на предположения за това, как знанията се запазват в човешкия мозък, а именнно – под формата на отделни единици, свързани помежду си с връзки. Семантичните мрежи могат да се разглеждат като вид логика, като основно тяхно предимство пред логиките като формализъм за ПИЗ е, че нотацията им обикнов е доста по-подходяща за графично представяне.
Има много варианти на семантични мрежи, но в същината си всички те представят знанията чрез обекти, категории обекти и връзки между обектите. Типичната графична нотация представя обектите и категориите в овали или правоъгълници и ги свързва с именувани дъги.
Под обекти в случая разбираме понятията от предметната област, а дъгите са връзките (взаимоотношенията) между тези обекти;
Б/ ТРОЙКИ "ОБЕКТ-АТРИБУТ-СТОЙНОСТ";
Модели с причинно-следствен връзки:
If условие, then действие
А1.Бинарни таблици.
А2. ПРОДУКЦИОННИ МОДЕЛИ - – продукционни правила.
Б. фрймови модели
В. ЛОГИЧЕСКИ МОДЕЛИ (ДЕКЛАРАТИВНИ)
Основният вид логика, който представлява база за повечето модели е предикатната логика от първи ред. Нейна основа е съждителната логика. За извличане на заключения в логическите модели се прилага дедукция. Важно е, че логическият извод се основава на строга математическа теория, гарантираща верността на резултата. Логическия извод няма ограничения за сложността на изразите.

19. Логически модели за представяне на данни. Предимста и недостатъци.

Математическата логика е от декларативния тип формализми за ПИЗ.
Основната цел на логиката е чрез средствата на формален език да се изрази знание за дадена предметна област. Логиката си служи с определени правила за извод. С тяхна помощ изразяваме и получаваме различни знания. Представянето на знания се осъществява чрез правилно построени формули в системата, а използването (извеждането) на знания се осъществява чрез правила и методи за извод. За логиката можем да си мислим като за език. В този ред на мисли има много начини да превеждаме от един език на друг. Каква част обаче от естествения ни език може да се преведе и отговаря на езика на логиката?
Логиката като всеки друг език се характеризира със синтаксис и семантика. Под *синтаксис* разбираме начина, по който конструираме изреченият, а под *семантика* – начина, по който интерпретира тези изречения. Нека разгледаме следния пример: „Всички лектори са високи 6 фута“. Това изречение само по себе си е напълно валидно от синтактична гледна точка. Освен това можем да разберем смисъла му, т.е. е и семантично валидно, но въпреки това изречението не носи вярна информация и го считаме за лъжа.
- Съждителна логика
- Логика от първи ред
- Логики от по-висок ред
- Логиките от по-висок ред са още по-описателни и изразителни от тази от първи ред. В логиките от по-висок ред се въвежда възможността за квантифициране на функции, предикати, а също и на обекти.
- Размитата логика е формализъм за ПИЗ, който произлиза от теорията на размитите множества и чиято цел е да отразява неточни определения от типа на: „много“, „малко“, „повечето“. Размитата логика е вид многозначна логика, защото за разлика от класическата логика, при която боравим с 2 стойности – истина и лъжа (1 и 0), то тук боравим с вероятност стойности, т.е. с променливи, остойностяв в интервала от 0 до 1. Под понятието многозначна логика разбираме такава, при която са допустими стойности, различни от традиционните - истина и лъжа.
- Немонотонните логики са апарат за ПИЗ, които се смятат за верни по подразбиране (по премълчаване).
Съществув различни начини за неформално въвеждане на понятието немонотонна логика. Един от тези начини е следният. Въвежда се логическата операция нормална импликация (Normal Implication, NI). Нормалната импликация P NI Q означава, че ако твърдението P е вярно, то твърдението Q също е вярно, освен ако по някаква причина не е известно, че Q не е вярно. Формалното въвеждане на нормалната импликация не е тривиална задача, тъй като при наличието на тази специална (в известен смисъл условна) импликация е необходимо да се въведат допълнителни правила, които позволяват да се вземе решение дали резултатите от изводите, които са направени с нейна помощ, могат да бъдат използвани за извършване на следващи такива изводи.
- Модалната логика е формализъм за ПИЗ, който разширява съждителната и предикатната логика, като включва оператори, които изразяват модалности от типа на „необходимо“, „възможно“, „случайно“. Например ако вземем твърдението: „Джон е щастлив“, то може да го разширим с модалност: „обикновено“ по следния начин: „Джон обикновено е щастлив.“. Освен модалностите към този формализъм се добавя и още едно понятие – логическата операция строга импликация. Тя се означава с „⇒“ и се дефинира по следния начин: p⇒q е еквивалентно на ¬M(p∧¬q), където с Mp означаваме модалността: p е възможно. От това можем да забележим, че истинността на строгата импликация не зависи от стойността на p.
- Темпоралната логика е формализъм за ПИЗ, който позволява представяне на знания чрез вземане предвид на времеви интервали. Например в темпоралната логика можем да изразяваме твърдения като: „Винаги съм гладен.“, „Ще съм гладен докато не се нахраня.“. Да вземем за пример твърдението: „Гладен съм.“ Въпреки, че значението на твърдението е константно във времето, истинността му може да варира във времето. Понякога твърдението е вярно, а понякога – не, но никога не приема стойностите истина и лъжа едновременно.
Точно с това се характеризира темпоралната логика – истинността на твърдението се променя във времето.

ПРЕДИМСТВА И НЕДОСТАТЪЦИ НА ЛОГИЧЕСКИТЕ СИСТЕМИ

Едно от предимствата на логическите системи като формализъм за ПИЗ е, че преводът от естествен към формален език е сравнително лесен, заради ясната им семантика и изразително. Има цели дялове в математиката, посветени на логиката, което я прави добър избор за представяне на знания в машинен вид. Освен това програмните езици сами по себе си са произлезли от логиката. В частност има програмни езици (като например Prolog), които използват предикатната логика, което значително улеснява представянето и използването на знания.

Най-съществените специфични предимства на логическите системи като формализ за ПИЗ са ясната им семантика и голямата им изразителна сила.

Значителни проблеми възникват при използването на логики, които са неразрешими или полуразрешими, а такива са всички логики с крайна аксиоматика.

Недостатък на този формализъм е, че възникват проблеми при използване на неразрешими или полуразрешими логики. Такива са всички логики с крайна аксиоматика.

20. Мрежови модели (ММ): Същност, видове, св-ва, наследяване, проблеми

с Graph DB може да се описват. Семантичните мрежи са типичен декларативен формализъ за ПИЗ. Знанията се представят с помощта на мрежа от възли и дъги, които ги свързват. С възлите се означават обектите (фактите, понятията и т. н.) от съответната предметна област, а с дъгите - връзките (отношенията, релациите) между тях. Често възлите (обектите) и дъгите (връзките) имат имена. По същество семантичната мрежа е орграф и следователно всички резултати и алгоритми от теорията на графите могат да се използват успешно при решаване на задачи, свързани със семантични мрежи.

Тази семантична мрежа представя знанията, които са описани чрез следните отношения:

През 1909 Чарлз Пиърс предлага графична нотация на върхове и дъги, наречена екзистенциални графи, която той нарича „логиката на бъдещето“. Освен това те принадлежат към т.нар. структурни формализми, които се характеризирт с това, че се основават на предположения за това, как знанията се запазват в човешкия мозък, а именнно – под формата на отделни единици, свързани помежду си с връзки. Семантичните мрежи могат да се разглеждат като вид логика, като основно тяхно предимство пред логиките като формализъм за ПИЗ е, че нотацията им обикновено е доста по-подходяща за графично представяне. Има много варианти на семантични мрежи, но в същината си всички те представят знанията чрез обекти, категории обекти и връзки между обектите. Типичната графична нотация представя обектите и категориите в овали или правоъгълници и ги свързва с именувани дъги. Под *обекти* в случая разбираме понятията от предметната област, а *дъгите* са връзките (взаимоотношенията) между тези обекти.

Типове дъги /връзки в СМ:

• тип “подмножество” (описват релации от тип клас – суперклас); • тип “елемент” (описват релации от тип обект – клас);

• тип “функция” (описват свойства на обектите и класовете).

Важно свойство на семантичните мрежи е възможността за обратни връзки. Възлите се използуват за представяне на обекти и описатели:

-Обектите могат да бъдат – физически обекти, концептуални действия, събития или абстрактни категории.

-Описателите дават допълнителна информаци за обектите.

Типове възли в СМ:

• релационни константи (таксономични категории или свойства);

• обектни константи (предмети или обекти от областта).

Връзките / дъгите свързват обектите и описателите. Връзките могат да бъдат (бинарни релации):

-“е” – взаимоотношение клас-екземпляр,

-“има” – взаимоотношение обект–свойство,

-“е част” - взаимоотношение част-подчаст,

-дефиниционна връзка – хората носят дрехи,

-причина – следствие и т.н.

В ЗАВИСИМОСТ ОТ ТИПА НА ВРЪЗКИТЕ МРЕЖИТЕ БИВАТ:

- *класификационни* мрежи – отношението е структуриране в различ йерархии
- *функционални* мрежи – описват процедурите за получаване на едни единици от други (изчислителни модели).
- *семантични* мрежи – допускат се различни видове връзки.

ПОНЯТИЙНИТЕ ГРАФИ са вид формализъм за ПИЗ, който спада към *семантичните* мрежи. Въведен е от Джон Сова през 1976г., като идеята му е да представя формули от предикатнат погика от първи ред чрез графи. Всеки граф представя дадено твърдение, като върховете на графа могат да бъдат както явни обекти (напр. човек, куче), така и абстрактни (напр. гняв, тъга). Дъгите при този вид графи не са именувани, вместо това са въведени върхове, които изразяват понятийни връзки. Основно предимство на този подход е, че връзката между много обекти се представя лесно чрез обект от тип връзка.

Други разновидности на СМ с които се разширява изразителната им сила (и се премахва недостатъка да представянето на квантите при предикатното):

• Представяне на произволни n-арни релации: *метод на Саймънс*;

• Представяне на знания, които се формулират чрез твърдения, съдържащи квантори за съществуване и всеобщност: разделени СМ (partitioned semantic nets).

а/ чрез въвеждане на разделените семантични мрежи (partitioned semantic nets). Семантич мрежа от този тип е йерархична съвкупност от пространства (подмрежи), всяко от които съответства на областта на действие на една или няколко променливи. Някои специални дъги в мрежата сочат не към възли, а към цели семантични мрежи, които са подмрежи на дадената. Тази семантична мрежа представя твърдението: “Кучето ухапа пощенс-кия раздавач.”. В нея възлите К, У и П представят съответно конкретно куче, конкретно ухапване и конкретен пощенски раздавач. Нека се опитаме да представим чрез семантична мрежа твърдението: “Всяко куче е ухапало някой пощенски раздавач.”. При намирането на представя на това твърдение чрез семантична мрежа съществено е построяването наподход представяне на областта на действие на променливата, пред която стои кванторът за всеобщност (която се отнася до квантора за всеобщност). За целта е целесъобразно да се използва механизмът на разделените семантични мрежи, с чиято помощ цитираното твърдение може да бъде представено чрез следната мрежа:

Възелът ТВ тук означава представяното твърдение като цяло. То е представител (частен случай) на специалния клас от Обобщаващи твърдения (ОТ) в предметната област. С други думи, ОТ е класът на Твърденията, които могат да бъдат записани чрез ППФ и съдържат поне един I. Всяко твърдение - представител на класа ОТ, има поне два атрибута: атрибут ФОРМА, който описва отношението, определено от твърдението, и един или повече атрибута V, които задават променливите, отнасящ се до I. В нашия случай има една такава променлива - К, която означава произволен елемент на класа КУЧЕТА. За останалите две променливи от ФОРМА (от мрежата S1) - У и П, се подразбира, че се отнасят

до i. С други думи, мрежата S2 представя следното твърдение: “За всяко куче К съществуват събитие на ухапване У и пощенски раздавач П, които са такива, че К е нападателят с У и П е жертвата от У.”.

б/ с използване на изкуствени конструкции (такъв е например известният метод на Саймънс), но така се нарушават посочените предимства на семантичните мрежи.

Същност на метода на Саймънс. Дадена n-арна релация се трансформира в система от бинарни релации чрез въвеждане на нов обект (нов възел в мрежата), който представя цялата релация. След това се въвеждат нови бинарни релации за описване на връзките между този нов обект и всеки от оригиналните аргументи. По този начин възлите в мрежата могат да представят не само обекти и понятия, но също така и ситуации и действия.

МЕХАНИЗЪМ ЗА ИЗВОД В СМ. При този вид представяне на знания се извършват изводи за обектите, така че да получим неявно зададени техни свойства.

Съществуват 2 основни механизма за извод върху семантичната мрежа:

1. *разпространяваща се активност / вълна* (търсене на сечение);

2. наследяване.

Виж 21 въпрос в детайли.

21. Механизъм на извод в мрежовите модели (ММ). Предимства и недостатъц на ММ.

Механизмът за извод има за цел да достигне до изводи за обекти като се намират стойности на техни свойства, които могат и да не са зададени в явен вид. Съществуват 2 основни механизма за извод върху семантич мрежа:

1. *разпространяваща се активност* вълна (търсене на сечение)

2. наследяване.

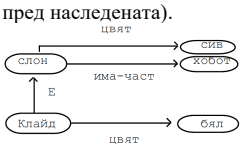
ПРИНЦИП НА РАЗПРОСТРАНЯВАЩАТА СЕ ВЪЛНА / активност. Този подход е средство за установяване на сходство в значенията на две думи – търсене на сечение. По същество е свързан с проверка дали в семантичните мрежи, които описват значенията на тези думи, се съдържа едно и също понятие (с други думи, подходът е свързан с *търсене на сечение на две семантични мрежи).*

Идея на подхода. Тръгва се по семантичните мрежи, които описват двете понятия. На всяка стъпка се достига до нови възли, чиито флагове на активност се вдигат. Движението (тъсенето) се прекратява или при достигане на възел, чийто флаг на активност е вдигнат (успех), или при изчерпване и на двете семантични мрежи (неуспех). Тръгва се от две понятия (например Уинстън и Китай). Отбелязва се, че те са активни (двата възела в графа). Отбелязват се дъгите, по които се разпространява активността. Активират се всички възли, с които е свързано първото понятие. Намират се възлите, свързани с второто понятие, и се проверява някой от тях не е ли вече активен. Ако е намерен активиран връх, то той е общ (сечението) между двете първоначални понятия. В противен случай се активират нови върхове, запомнят се дъгите и т.н. докато се изчерпи мрежата. (мрежите, ако понятията са от две несвързани мрежи.)

ПРИНЦИП НА НАСЛЕДЯВАНЕТО: Обектите и класовете могат да наследяват свойства от класовете и суперкласовете, към които принадлежат. Причисляването към даден клас или суперклас обикновено става с помощта на връзки с примерни (най-често срещани) имена Е (бълг.), ISA (ISA на англ., Е на бълг.), Inst (instance), Sub, Superclass и др. Тук са възможни два подхода:

a. връзките обект - клас и клас - суперклас да имат едно и също име (например ISA);

b. връзките обект - клас и клас - суперклас да имат различни имена (например ISA и Superclass или Inst и Sub). По отношение на наследяването на свойства няма значение кой от двата подхода ще се избере. Механизъм на наследяването. Ако се интересуваме от определено св- во на даден обект (или клас), най-нап ед п ове яваме дали същото свойство е зададено в явен вид за нашия обект (клас). Ако това е така, извличаме явно зададен стойност на свойството; ако не, тръгваме по пътя, сочен от връзките от типа на ISA, и търсим стойност на интеесуващото ни свойство, зададена за някой от класовете, към които нашият обект (клас) принадлежи. В този случай се казва, че обектите наследяват свойствата на класовете, към които принадлежат. По този начин лесно се извършва извод с използване на знания, верни по премълчаване (ако дадено свойство е зададено в явен вид, явно зададената стойност е с приоритет пред наследената).



Ще направим някои бележки върху целесъобразността от използване на различни връзки за изразяване на релациите обект - клас и клас - суперклас.

Нека е дадена следната семантична мрежа, в която всички релациите обект - клас и клас - суперклас са изразени с една връзка Е.

Ако искаме да зададем въпрос: “Колко (кои) животни познавате?”, възмож-но е да получим следните отговори:

три (котка, куче, кон) - ако се търсят имената на възли, непосредствено свързани с понятието животно с връзката Е;

пет (Кора, Том, Рекс, Лорд, кон) - ако се търсят листата на дървото.

Ако искаме да получим отговор четири (Кора, Том, Рекс, Лорд), съответен на броя на конкретните обекти (а не класове), трябва да въведем два типа връзки, например Е (обект - клас) и Суперклас (клас - суперклас).

ПРЕДИМСТВА И НЕДОСТАТЪЦИ НА СЕМАНТИЧНИТЕ МРЕЖИ. Специфични *предимства* на семантич мрежи: графите се представят много лесно в паметта на компа. Семантичните мрежи се представят чрез графи, което означава, че всички алгоритми от теорията на графите могат да бъдат използвани за решаване на задачи със семантични мрежи. Освен това семантичните мрежи използват голяма част от естествения език за представяне на знания, при това ако две изречения с еднакви значения бъдат представ с графи, то графите им също ще са еднакви. Техни съществени предимства остават:

-простота, яснота и естественост от човешка гледна точка;

-приложимост за всякаква предметна област;

-гъкавост, която се проявява в лесно добавяне и отстраняване на възли и връзки;

-възможност за наследяване, с което броя на записите може значително да се съкрати;

-наличие на организационни оси за структуриране на БЗ: класификация – всеки обект е свързан със своите части или компоненти, обобщение – свързва един тип с други родове и типове, разделение – групиране на обектите и отношенията в раздели, както са организирани йерархично;

-възможности за развитие, но всяко предложено решение накърнява някои от основните им предимства: простота, естественост за човека, нагледност, яснота за човека.

Основ. НЕДОСТАТЪЦИ на този формализъм е, че С помощта на семантични мрежи се изразяват удобно само бинарни релации, тъй като дъгите на мрежата съответстват само на бинарни релации между възлите. Това означава, че изречения от вида: Fly(Shankar, NewYork, NewDelhi, Yesterday) не могат да бъдат директно вкарани в семантична мрежа. Специфичен недостатък е и че наследяването, извършван на различни операции и самото реализиране на такива системи се управляват трудно. *Трудно се представят знания, които съдържат квантори.* Семантичните мрежи имат недостатъчна изразителна сила в сравнение с предикатното смятане - например с помощта на класическите семантични мрежи трудно се изразяват знания, които съдържат квантори - недостатък спрямо предикатната логика от първи ред е, че връзките между различните обекти

в семантичните мрежи са единствено бинарни (т.е. могат да се изразяват само бинарни взаимоотношен между обектите и знания които съдържат квантори). За целта се ползват и внедряват разновидности на СМ с които се разширява на изразителната им сила:

• Предствяне на произволни n-арни релации: метод на Саймънс;

• Представяне на знания, които се формулират чрез твърдения, съдържащи квантори за съществуване и всеобщност: разделени СМ (partitioned semantic nets).

а/ чрез въвеждане на разделените семантични мрежи (*partitioned semantic nets*). Семантич мрежа от този тип е йерархична съвкупност от пространства (подмрежи), всяко от които съответства на областта на действие на една или няколко променливи. Някои специални дъги в мрежата сочат не към възли, а *към цели семантични мрежи*, *които са подмрежи на дадената.*

б/ с използване на изкуствени конструкции (такъв е например известният *метод на Саймънс*), но така се нарушават посочените предимства на семантичните мрежи.

Същност на метода на Саймънс. Дадена n-арна релация се трансформира в система от бинарни релации чрез въвеждане на нов обект (нов възел в мрежата), който представя цялата релация. След това се въвеждат нови бинарни релации за описване на връзките между този нов обект и всеки от оригиналните аргументи. По този начин възлите в мрежата могат да представят не само обекти и понятия, но също така и ситуации и действия.

Неясна семантика като недостатък на СМ. семантиката му понякога е неясно дефинирана.

Значенията, присвоени на върховете, могат да бъдат двусмислени.

Въпреки името си семантичните мрежи наистина понякога имат неясна семантика. Това означава, че една семантична мрежа често може да бъде тълкувана по различни начини и следователно е необходимо въвеждането на съответна еднозначна семантика. Проблеми при извършването на различни операции със семантични мрежи. При обединяване на семантични мрежи може да се дублира информация за съвпадащи обекти - известен под името "проблем на двамата Смита." Ако в мрежите съществуват различни обекти с еднакви имена (например хора с еднакви собствени имена), необходимо е или съответните възли да се преименуват (в случая например от име да се премине към име и фамилия), или да се промени структурата на мрежата, като например се въведат нови възли В такъв случай дублиращите се елементи трябва да се премахнат и съответните възли и връзки да останат в един екземпляр. *Проблеми при управлението на наследяването.* Не винаги е целесъобразно да се извършва наследяване на свойства, а трудно се задава частична забрана за наследяване. Пример за неподходящо наследява. Даден човек има вкъщи като лична собственост птичка, която е от рядък вид - обект на защита от закона и обект на научни изследвания. Тези две различни свойства са свойства на вида и не би трябвало непременно да се наследяват и от конкретния индивид.

Обща оценка на семантичните мрежи като формализъм за ПИЗ

Специфични преимущества:

простота, нагледност, яснота, естественост на представянето.

Недостатъци и проблеми:

• недостатъчна изразителна сила;

• неясна семантика;

• проблеми при извършването на различни операции със семантични мрежи;

• проблеми при управлението на наследяването.

22.Продукционни модели. Общ вид на продукция. Видове ядра на продукция.

Това е декларативен формализъм с елементи на процедурност на по-ниско равнище. Негова основна характеристика е декомпозирането на знанията на малки части (*правила*) от тип условие - следствие (ситуация - действие). Т.е. знанието се представя чрез двойки от вида: <условие; следствие>. Тези двойки образуват множество от правила, определящи действието на системата. За всяка такава двойка агентът проверява дали усл-то е издържано и ако е така, продукционното правил задейства следствието и то се изпълнява. Продукциите отразяват релацията по най-близък за човека начин. Те са относително близки до логическите модели и дават възможност да се организират процедури за извод. В същото време продукционните модели не са така формализира и ограничени както логическите модели.

Всяка продукция може да се разглежда като правило за извършване на определени действия.

ОБЩ ВИД НА ПРОДУКЦИЯ. Знанията са представени в удобен вид.

(i) Q; P; A => B; N;

Първата съставка i е име на продукцията. То я отделя от останалите продукции. Името може да бъде:

- число (например пореден номер в базата знания); - дума или набор от думи, които отразяват същността на продукцията. (например "търсене на път в дървовиден граф").

Втората съставка Q е област за приложение на продукцията.

Знанията се групират по сфери на приложение. Това дава възможност да се осъществи по-ефективно търсене в базата от знания. За да се осъществи по-ефективно търсене, се използува областта за приложение на продукциите.

Третата съставка P е условие за приложимост на продукцията. Често това условие е в предикатна форма. Продукцията може да се използва, само ако това условие е истинно. P – логическо условие (предикат); If A then B

A – условие

B – следствие.

If A then B else C

Четвъртата съставка A=>B е ядро на продукцията. Знакът =>се интерпретира като "следва". Ядрото може да има различни форми. Най-често:

АКО A ТО B (if A then B). Възможни са и други варианти, например:

АКО A ТО B1, в противен случай B2 (if A then B1 else B2). Ядрото може да се интерпретира по различен начин. Например:

-логическо интерпретиран – ако условието е истинно, то истинно е и B; ако условието не е истинно, то за истинността на B нищо не може да се каже.

-A е условие за изпълнението на действието B.

По горните бяха прости ядра. Едно сложно ядро може да има следния вид:

Ако A1,A2, … , An , то (B1, B2, … , Bn)

И от двете страни смисълът е конюнкция.

Последната пета съставка N е *постусловие* на продукцията. Постус-то се активира след като продукцията се изпълни.

Постусловията описват действия и процедури, които трябва да следват изпълнението на продукцията. Постусл-то е след указанието, какво да се прави.

1.Прави се списък на всички продукции, чиито активиращи условия са изпълними.

2.Избира се една от тези продукции.

3.Изпълняваме тази продукция

ВИДОВЕ ЯДРА на ПРОДУКЦИЯ –

Ядрата могат да бъдат детерминирани или недетерминирани:

При детерминираните ядра ако условието е изпълнено, дясната част се изпълнява обезателно.

При недетерминираните ядра са възможни варианти. Ако A, то е възможно B.

Възможността може да се изрази по различен начин, например чрез:

- вероятност

Ако A, то с вероятност p да се реализира B.

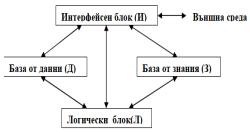
- семантична оценка за възможност

Ако A, то с увереност да се изпълни B. почти сигурно и т.н. Възможна е и алтернативност:

Ако A, то по-често B1, а по-рядко B2. В такива случаи са необходими тегловни коефициенти за избор (вероятностна оценка, лингвистична оценка, експертни оценки). Някои продукции могат да представляват прогнозира продукции. Например

Ако A, то с вероятност p може да се очаква B.

Източник и приемник на информацията за продукцията би могъл да бъде всеки от блоковете на следната схема.



Възможните варианти личат от следната таблица.

В\А	И	Д	З	Л
И		*		*
Д	*	*	*	*
З	*	*	*	*
Л	*		*	*

Записът Ах => Ву означава, че информаци се взема от блок х, а резултатът от действието В се изпраща в блок у.
Например АД => ВД.
Най-често АЗ => ВЗ - замяна на един фрагмент от базата знания с друг фрагмент (получен чрез ВЗ). За да се задейства тази продукция е необходимо в БД да бъде намерен фрагмента АЗ. (търсене по образец).
АД => ВЗ – намиране на закономерности по емпирични данни.
Всяка система с изкуствен интелект, основана на правила има 3 основни компонента –:
1. *РАБОТНА ПАМЕТ (РП)* (контекст) – съдържа входните данни за задачата известни към текущия момент. Те могат да бъдат получени от 2 източника: както от *потребителя*, така и от *интерпретатора* в процеса на логическия извод. Представянето на данните в работната памет зависи от синтаксиса на правилата, но обикнов е под формата на тройки от вида:
<обект;атрибут;стой-ст> (обектът има атрибут със зададената стойност) или <атрибут;релация;ст-ст> (атрибутът се намира в дадена релация с конкретната стойност).
2. *БАЗА ОТ ПРАВИЛА (БП)* – съвкупността от правилата, които са ни дадени за конкретната предметна област. Тук се пази относително постоянната част от данните на с-мата. Общ вид на правилата
(if <условие1> <условие2>... <условиен >
then <следствие1> <следствие2>...
<следствиеm >)
Обща идея за семантика. Ако са верни условията, то са верни и следстви-
ята (или да се изпълнят следствията).
Предполага се конюнкция между условията. Ако е необходимо да се изрази дизюнкция, обикновено това става с използване на отделни правила.
Условията често се наричат лява страна на правилото, а следствията дясна страна. 1. Предполага се конюнкция между условията. Ако е необходимо да се изрази дизюнкция, обикновено това става с използване на отделни правила.
2. Условията често се наричат лява страна на правилото, а следствията - дясна страна.
Съществуват много различия между различните ПС (прод. с-ми) в зависимост от точния синтаксис на правилата:
- По отношение на допустимите следствия (действия) в дясната страна: в по- простите системи се допуска само включване на нови елементи към РП, а в по-сложните - и изключване и изменение на елементи от РП.
- По отношение на допустимостта на използването на променливи в лявата и/или дясната страна: в най-простия случай не се допуска използването на променливи. Тогава правилата имат малка изразителна сила и в процеса на работа на интерпретатора е възможно единствено сравняване на съответните условия с елементите на РП с цел откриване на съвпаден, т. е. не се извършва съпоставяне в истинския смисъл на думата. Ако се използват и променливи, тогава правилата имат много по-голяма изразителна сила (могат да изразяват правила за извод, свързани с обобщения). В този случай е необходимо съпоставяне на съответните условия с елементи на РП и в резултат може да се извърши свързване на променливите от правилото с константите, с които те са били съпоставени.
-По отношение на броя на допустимите елементи на лявата/дясната страна: от гледна точка на работата на интерпретатора има значение дали в лявата част на правилата се допуска наличие само на едно или на произволен брой условия.
3. ИНТЕРПРЕТАТОР (машината за извод на съответната система) – това е програмна с-ма, чиято основна цел е да прилага въведените в базата правила върху данните от работната памет и да изведе ново знание. Работата на интерпретатора се разделя на две части – *избор на правило и изпълнение (интерпретация) на правилото.* Ро същество задачата в първата фаза се свежда до търсене в БП по някакъв образец. Според образеца на търсене, а следователно и според начина на интерпретация се различават два типа системи от продукции (СП) - прави (F_продукции, Forward - напред) и обратни (B-продукции, Backward - назад). Така интерпретат може да реализира един от следните 2 вида алгоритми:
<u>алгоритъм за прав или за обратен извод</u>.
ПРАВИЯТ ИЗВОД се нарича още извод, *управляван от данните*, а обратният – извод, *управляван от целите.*
Прав извод – при този подход интерпретаторът търси правило, чието условие се удовлетворяв от данните в контекста, след което изпълнява следствието от двойката, чрез която е представено правилото. Ако съществ обаче повече от едно правило, което удовлетв даденото условие може да възникне конфликт. Тогава всички, удовлетв условието правила, образуват конфликтно множество и целта ни е да изберем едно от тях, което да изпълним, т.е. прилагаме т.нар. *конфликтна резолюция*. Има различни подходи за избора на правило – например първото срещнато правило, остойностяване на правилата с тегла или приоритети или правило, което не е използвано до момента и др.
ОБРАТЕН ИЗВОД – при този подход – алгор. интерпретат извършва търсене върху десните страни на правилата (т.е. върху следствията). След това се прави проверка дали условието в лявата част на правилото удовлетворя зададената цел. Това се прави, докато се достигне зададената цел или докато бъдат изследвани всички възможности.

23. Продукционни модели – основен обработващ цикъл. У-ние на с-ма от продукции – стратегии. Предимства и недостатъци на ПМ.

ОСНОВЕН ОБРАБ. ЦИКЪЛ
БД съдържа конкретни стойности за всички величини- характерист на обектите, които са достъпни за всички продукции както при проверка на стойността, така и за нейното изменение.
Обработката следва следната схема:
1) Проверка за изпълнение на активиращите условия (левите страни) на всички продукции.
Определя се множеството продукции, които могат да се изпълнят. Те се конкурират помежду си.
2) По определена стратегия се избира една от конкуриращите се продукции. (Всички останали продукции не се изпълняват).
3) Изпълнява се избраната продукция като с това се променя състоянието на базата от данни. Ако не е налице условието за край следва отново 1.
При изпълнението на първата стъпка от обработващия цикъл се открояват два момента:
- *обхождане на продукциите в базата знания* и
- *търсене на стойности* в базата данни.
От рационалното им организиране зависи ефективността на системата.
Стратегия за избор на продукция, която трябва да бъде изпълнена.
<u>Стратегии за обхождане на продукции.</u> В това отношение възможните варианти са много и ще спомена само два примера.
1)Използува се следната идея – най-често използваната продукц е най-полезна. За да работи успешно тази идея, необходимое продукциите да бъдат независими една от друга. Те се подреждат в базата знания по честотата на използване и всеки път се обхождат в този ред. Ако се добави и обратна връзка по “успешно прилагане” на продукциите се получава обучаваща се система, която се усъвършенствува в процеса на работата си.
2)Принцип на най-дългото условие. Този принцип се основава на “здравия смисъл” - частните случай са по-полезни, тъй като отчитат по-голям брой условия. За да работи този принцип, необходимо е подреждане на продукциите в реда от по-частни към по-общи. Търсенето в базата данни

отнема значително време, поради което е необходимо да се предприемат мерки за рационалното му организ. Ето някои от използуваните вариант за организиране на този процес:
- последователно търсене; има най-ниска ефективност;
- търсене чрез индексиране на базата данни;
- разбиване на подбази от данни и въвеждане на правила за избор на подбаза.
- разбиване на продукциите на групи по област на приложение и формиране на база данни за всяка подгрупа;
- вътрешно структуриране на БД и проверка само на онези правила, в чиито условия съществуват обекти, променени в предходната итерация и т.н.

За <u>оптимизиране изпълнението</u> на първата стъпка могат да се използуват следните идеи:
* Много продукции имат общи условия или части от условия. Следователно проверката на тези условия може да се извър само един път.
* При изпълнение на продукции само малка част от данните се променят. Следователно може да се проверява само променилата се част от данните, тъй като резултатите от проверката на другите данни са както при предишната итерация.

При изпълнението на втората стъпка от обработващия цикъл се използуват разнообразни стратегии за избор на продукция. Най-често стратегиите за избор на продукция са известни предварително и са твърдо заложени. Използуват се следните стратегии за избор:
- по номер-избира се продукцията, която има най-малък номер (например пореден номер)
- по "възраст" на данните - избира се продукцията, чиито активиращи данни са били най-скоро променени, или които тя ще промени.
- по успешно участие - избира се продукцията, за която водената статистика показва, че най-често е използувана при довело до успех търсене на решение. За изпълнението на третата стъпка ще спомена само, че възможността за паралелно изпълнение на всички активирани продукции може да доведе до случай, когато две от тях изменят едни и същи фрагменти от БД по различен начин. Необходимо е такива конфликти да се откриват и да се решават, което може да намали ефективността от паралелното изпълнение в значителна степен. Когато две или няколко продукции изменят едни и същи данни по различен начин, те образуват конфликтно множество продукции. *При избор на продукция от конфликтно множество продукции* може да се постъпи по някой от следните начини:
- избира се първата срещната продукция;
- избира се продукцията, която при този извод не е използвана.

Конфликтните множества от продукции могат да се намалят чрез добавяне на нови условия. Но тогава много от нововъведените продукции могат да се окажат взаимнозависими

У-НИЕ НА С-МАТА ОТ ПРОДУКЦИИ – СТРАТЕГИИ.

При управлението на система от продукции трябва да се отчитат следните възможности:
- липсват предписания за реда за изпълнение на продукциите;
- има изисквания към реда за изпълнение на продукциите; В този случай е необходима информация коя продукция след коя ще следва.
- смесен вариант; В този случай за някои продукции е определен ред за изпилнение. Като частен случай в постусловието се посочва името на продукцията, която трябва да се изпълни като следваща.

Продукциите могат да бъдат управлявани
- *Централизирано* чрез изградена система за управление на избора.
- *Децентрализирано* от текущото състояние на БД.

При работата със система от продукции се използуват разнообр техники.

<u>*Използване на метапродукции.*</u>

Метапродукциите се изпълняват първи и при активирането си определят реда за изпълнение на останалите продукции.

<u>*Принцип на "дъската за обяви".*</u> В оперативната памет се оформя специална област, наречена "дъска за обяви". Изпълняваните продукции четат "обявите" от черната дъската и записват резултатите от своята работа също върху нея.

<u>*Принцип на приоритетния избор.*</u> За целта въз основа на важността на продукциите за дадена предметна област за всяка от тях се задава приоритет. Този приоритет се нарича статичен приоритет. Подреждането на продукциите по важност не е лека работа и се предлага от експертите. За усъвършенствуване управлението на системата се създава механизъм за поддържане на динамич приоритети, които се изработват в процеса на работа на системата.

<u>*Управление от имената.*</u>

За имената на продукциите е зададена система от правила (например граматика), която стеснява кръга на активираните продукции, които трябва да се изпълняват.

ВИЖ предния въпрос! В продукционните системи изводът може да бъде реализиран в два варианта - *прав извод (forward chaining*) и обратен извод (*backward chaining),* които се наричат още съответно:
-извод, управляван от данните (data-driving inference) -извод, управляван от целта (goal-driving inference)

Нека повторим правия извод:
1. Търсят се продукции, които могат да се задействат при дадено състояние на БД .
2. Избира се една от тях.
3. Изпълнява се тази продукция и ако не е достигната до целта отново се преминава към

1. Обратен извод:
1. Намират се продукциите, които имат за дясна част търсената цел.
2. Избира се една от тях и параметрите от лявата й част, които не са намерени в базата данни се формират като нови цели (подцели).
3. Действа се така докато се стигне до удовлетв от БД на всички формирани подцели или се изясни, че това е невъзможно. При неуспешен край може да следва връщане назад (backtracking) и избор на ново правило, чиято дясна част е търсената цел.

ПРЕДИМСТВА И НЕДОСТАТЪЦИ

Основно ПРЕДИМСТВО на този формализъм е естествеността на представяне на знанията. Друго предимство е, че при този формализъм има ясно изразена модулност (модулна ст-ра) – т.е. *постоянните знания* са отделени от временните (първите се пазят в базата от правила, докато вторите са в работната памет). Тази модулност води до:
- лесно изменяне на правила в системата.
- сравнително лесно се откриват и отстраняват грешки в базата от правила, заради структурата на всяко правило – две независими части (условие и следствие). Ако правилото се изпълнява в неподходящия момент, лесно се открива условието, поради което това се случва, а ако се изпълнява неподходящо действие, лесно се открива правилото, чието следствие е даденото действие.
- по удобен и естествен начин се извършва изграждането на БП на части. Това е особено важно при създаването на ЕС. Авторът є може най-напред да се съсредоточи върху изграждането на тази част от БП, която позволява на системата да решава дадена конкретна задача, и едва след това да се опитва да допълва и обобщава БП за други задачи;

Към *ПРЕДИМСТВАТА* на продукционните системи могат да се отнесат: - Системите продукции се използват широко при изграждането на експертни системи (ЕС).
- Те са удобен начин за представяне на човешките знания.
- Притежават естественост при представяне на експертно знание.
- Системата е модулна и с редки изключения отстраняването и добавянето на продукция не изменя останалите.
-Всяка продукция е самостоятелна, независимо от останалите знания. Създаването и актуализацията се извършват лесно. Върху БЗ могат да работят едновременно няколко души.

- Налице е ясна идея за реализиране на обяснения.
- Поддават се в повечето случаи на структуриране.
- Съчетават се много добре с мрежовите модели.
- Наличие на естествен паралелизъм и асинхронност, което е от значение за новите поколения компютри.
- Простота на системите, които редактират знанията, плюс възможността за създаване на програми, които извличат (чрез "разпитване") знания от експертите.
- огр. синтаксис, който води след себе си предим че лесно се правят програми – редактори за СП в което се редактират праивлата и се изгражда и поддържа БП; лесно се реализират средств. За ген на обясненията на ЕЕ; лесно се изгр програми с които се извилза знания от хора – експерти или ЕС; Възможност за обяснение на взетите решения като предимство. Наличието на средства за обяснение на взетите решения е особено важно при консултиращите системи (консултиращите ЕС, основани на СП).

Основен *НЕДОСТАТЪК* на продукционните правила като формализъм за ПИЗ е проблемът за обясняването на взетите решения (на основата на системата MYCIN). За разлика от човека, който лесно дава обяснение защо е избрал дадена хипотеза и след това друга, при системите с продукционни правила това не изглежда по този начин. За преодоляване на този проблем е необходимо разширяване на съществуващата система от правила или използване на изцяло друг подход.

Друг *недостатък* е и ниската ефективност при избор на правило от конфликтното множество в случай на възникване на конфликт, както и при работа с големи системи.

Източници на тази неефективност са двете основни операции при определянето на правилото, което трябва да се изпълни от интерпретатора на правилата:

определяне на конфликтно множество от правила (които могат да бъдат изпълнени на дадената стъпка от работата на интерпретатора);

избор на правило от конфликтното множество, което да бъде изпълнено (конфликтна резолюция).

Немаловажен недостатък е и че не всички знания могат да бъдат представени под формата на правила. Към по-съществените *НЕДОСТАТЪЦИ* на продукционните с-ми спадат още:

-При голям брой продукции проверката за непротиворечивост се затруднява и забавя;

- Съществени са и трудностите за проверка дали системата работи коректно (възможностите за избор на една от многото активирани продукции).
- Огр синтаксис от правилата. Дезюнктивни знания - В лявата част не може да се поставят дизюнкции if (a ∨ b) then q. В този случай трябва да се запишат две независими продукции:

if a then q

if b then q

или отрицателни знания - В условията на правилото могат да се изразят знания, които съдържат логическо отрицание, но не и в следствията му.

24. Фреймови модели - идея и същност. Фрейм - структура и съдържан. Видове фрейми.

ВЪВЕДЕНИЕ: При фреймовете са съчетани декларативни и процедурни знания. Част от знанията са включени в статична декларативна структура фрейм. Освен това има и допълнителни знания (обикновено процедурни) за това, как да се използват описателните (деклерат.) знания от фрейма, как да се свързват тези знания със знанията от други фреймове и т. н.

Едно от структурните представя на знания и се базира на предположението, че в човешкия мозък знанията се съхраняват на "порции" – обособена група от обекти и отношения, обединени около конкретно понятие или ситуация. Тази постановка лежи в основата на т.н фреймови модели за представяне на знанията (от frame-рамка). Фрейм – порция от знания, свързани с нашите знания. За първи път тази идея за представяне на знания се въвежда от Марвин Мински през 1975г. Той дефинира понятието фрейм (рамка) като „структура от данни, предназначена за описание на дадена стереотипна, стандартна ситуация". Основна характеристика на фреймовете е, че описват знания, които винаги са верни и структурата на тези знания и връзките между отделните такива е заложена в самия фрейм.

Накратко: фреймът съдържа информация за това, което би се случило в следващия момент и какво да се направи, ако това се случи или не се случи. Характерно за фреймът като структура е, че:

- описва знания, които са общи и винаги верни;
- структурата на фрейма е следствие от структурата на знания и връзките между тях.

Пример (Мински): „Преди да влезе в дадена стая, човек вече е подготвил в съзнанието си образа на стая „в общи линии" – стени, таван, под, врата, прозорци и т.н. Когато влезе в стаята, той само допълва и конкретизира своята представа за стая с особеностите на конкретната стая." На фреймовете може да се гледа и като на образци с определени особености, които конкретизираме за съответната ситуация.

Малко по-късно Шенк предлага фреймът да се използува и при описването на сценарии - конкретна типична поредица от събития. Пример за фрейм-сценарий: изпит – книжка със заверен семестър, влизаш, теглиш билет, сядаш, развиваш въпроси, решаваш задачи, отиваш при преподавателя, говориш, стараеш се да заобиколиш това, което не знаеш, получаваш оценка и т.н. Всъщност фреймите се появяват като апарат за използуване на обобщ. описания на обекти и ситуации и дават възможност представата да не се изгражда изцяло всеки път, а да се използуват готови обобщени структури (празен фрейм - прототип), които само се попълват и конкретизират. Така новите знания се изграждат в контекста на наличните (на основата на натрупания опит).

Накрая, при фреймите се съчетават *декларативни (описателни)* знания с *процедурните (алгоритмични)* знания. В един фрейм могат да фигурират както директ стойности на дадена величина, така и начини, по които тези стойности могат да бъдат намерени.

СТР-РА НА ФРЕЙМ, Фреймът представлява структура, която се състои от *име* и няколко полета, начени *слотове*. Всеки слот има име и съдържание. Името на слота показва за какво се отнася този слот. Слотове - това са свойствата на описвания обект, като свойствата носят и стойностите, които заемат. Всяка стойност може да бъде: стойност по подразбиране; наследена стойност от фрейм на по-високо ниво в йерархията; процедура, наречена демон, която служи за намиране на стойността; специфична стойност, която може да служи за представяне на изключение от типичните за този слот стойности. Стойностите във фреймовете на по-високо ниво могат да бъдат интервали от стойности, а също и условия. На по-ниските нива стойностите могат да бъдат конкретни стойности, които да пренаписват стойностите, наследени от по-високи нива. Слотовете във фреймовете могат да съдържат следната информация: информация за избора на фрейм в дадена ситуация; информация за връзките на този фрейм с останалите; процедури, които да се изпълнят след като определени слотове са запълнени; информация по подразбиране, която да се използва в случай на липсваща (непопълнена) информация; др. фреймове (така се реализира йерархичната структура). Съдържанието на слота може да бъде:

-числова стойност;
-текстове на естествен език;
-математически зависимости;
-процедури;
-правила за извод;
-препратки към други фрейми и т.н.

Формално това може да се запише така:

(Име на фрейма:
Име на слот 1 (съдържание на слот 1);
Име на слот 2 (съдържание на слот 2);

……
…….
Име на слот N (съдържание на слот N))
Пример:
(Студент:
Име (Антуан Ангелов);
Роден (05.11.1970);
Изпити (изпити);
Успех (процедура);
Стипендия (ако “успех” >= 5,00 то стипендия (40/ 5) * успех)).
(Изпити:
математика (6.00);
физика (3.50);
изкуствен интелект (6.00)).
Към един слот могат да се прикрепят няколко т.нар. *фасети*. Фасетите се използуват за описване на допълнителни данни относно описваната в слота величина. Например: Фасетата “стойност” (*value*) посочва конкретна стойност на описваната в слота величина. Фасетата “стойност по подразбиране” (*default*) съдържа стойност, която се използува, ако във фасетата “стойност” не е посочено нищо. Фасета “*if needed*” с име и тяло на процедура, която се стартира, ако първите две фасети са празни, и изчислява стойността на величината. Фасета (*if added or if removed*) с процедури, които се задействат автоматично, ако бъде добавена(if added) или изтрита (if removed) стойност на съответната х-ка. Процедурите, които са прикрепени към даден слот, могат да бъдат от два типа. Първият тип са процедурите - *демони*, които се активират автоматично при определени условия. Процедури от този тип могат да бъдат прикрепени и към фрейма като цяло. Те следят и евентуално сами изменят част от знанията във фрейма или предизвикват изпълнението на по-сложни операции върху цялата мрежа от фреймове (създаване или унищожаване на екземпляр на фрейм, промяна на друг фрейм и др.). Вторият тип са процедурите - *методи*. За разлика от демоните процедурите - методи не се изпълняват автоматич, а се задействат или от системата за управление на фреймовете, или от др процедури, прикрепени към фреймовете. Те следят непрекъснато за достъп до слота, към който са прикрепени, и се задействат при конкретни случаи. От гледна точка на процеса на *извод* при фреймовите модели особена роля играе слота “от клас” *(A kind of) – а-к-о*. В този слот са зададени имената на фреймите, които описват класовете, към които се причислява обектът (или подкласът, описван от конкретния фрейм). Тази конструкция може да се съчетае с някои от посочените по-горе фасети. Фреймовите модели могат да обединят всички основни особен на останалите модели за представяне на знания.

ВИДОВЕ ФРЕЙМИ

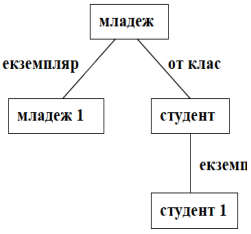
Като начало нека разделим фреимите на 2 групи – фрейми прототипи и фрейми екземпляри Фреймите прототипи задават структурата, чрез която се описват всички екземпляри от даден клас или подкласове. Фреймите екземпляри съдържат стойностите за всеки конкретен елемент от класа (или подкласа).
Например:
Пр. за фреим прототип:
(*Моторно превозно средство:*
регистрационен номер (стойност на слот 1);
вид регистрация (стойност на слот 2);
рама (номер на рамата);
двигател (номер на двигателя);
марка (заводска марка);
модел (стойност на слот 6);
вид (стойност на слот 7);
брой места (стойност на слот 8);
цвят (стойност на слот 9);
дата на регистрация (стойност на слот 10);
собственик(трите имена и адрес на собственика);
последен технически преглед (дата)).
Пр. *фреим екземпляр (Моторно превозно средство:*:
рег. N:(С 4242АВ);
вид рег. (Ф);
рама (011123528);
двигател (152538425);
марка (ВАЗ 2137);
модел (комби);
вид (лек);
брой места (4+1);
цвят (зелен);
дата на рег. (05.02.1987);
собственик (Иван Петров Добрев, София, Криволак, 12);
последен технически преглед (15.06.1990)).
Фреймът екземпляр се получава от фрейма прототип като последователно се запълват стойностите на слотовете. При създаването на фрейма може част от слотовете и фасетите да нямат стойност.
Примери за клас-подклас:

Примери за клас-подклас:

Прототип	Екземпляр
Младеж	Младеж 1
A kind of (от клас хора)	A kind of (от клас хора)
Име(Име,фамилия)	*име (Бранимир Петров)*
пол	*пол (мъжки)*
възраст (>16)(<35)	*възраст (23)*
интереси(с по една дума)	*интереси (тенис, Heavy Met*

Студент	Студент 1
A kind of (от клас младеж)	A kind of (от клас младеж)
Университет(Име)	*университет (ТУ-София)*
Специалност()	*специалност(Изчислителна*
Име(Име,фамилия)	*име (Мария Иванова)*
Пол	*пол (женски)*
Възраст	*възраст (19)*
Интереси(с по една дума)	*интереси (музика, плуване)*

Фреймите биха могли да бъдат подредени в структури. Екземплярит от един подклас наследяват всички слотове, които са дефинирани от фреймите-прототипи за класове над тях.



Фреймите се използуват за описание на както на данни, така и на правила! В такива случаи те могат да се класифицират по следния начин:
- *фрейми-данни*
М 6-3:
Представлява (кладенец);
Снабдява (М6-2);
Разположение (Главна улица);
- *фрейми-правила*.
Правило #332:
Представлява (правило);
Същият (съобщава за опасност);
Ако потенциално (Висока химическа активност)
Ако действително (Химически източник наблизо)
То: Да се каже на потребителя да не диша
То: (Да се добавят допълнителни указания)
Приоритет (висок)
Средно време за изпълнение (0,1 s)
Честота на използване (за 985 пъти използвано 4 пъти)

25. Извод при фреймовите модели. Стратегии при наследяване на св-ва. Предимства и недостатъци на ФМ.
При работа с фреймови модели от съществено значение е проблема за истинност на фреймите в проблемната област.
Използуват се основно 2 *модела*:
А/ *Затворен модел*. При затворения модел фреймът, който се съхранява в базата, е абсолютно истинен, а фреймите, които не са в базата, са абсолютно неистинни. Такава база се нарича затворена (например: полети на дадена авиокомпания). Затвореният модел е твърд ограничен и макар и удобен за изпълнение и работа се използува рядко.
Б/ *Отворен модел*. При отворения модел фреймите в базата се съхраняват с указание за тяхната истинност или неистинност, а фреймите, които са извън базата,се смятат за такива с неопределена истинност. *Терминална стойност* е такава стойност, която не може да се разложи (или не се разлага) на по-прости стойности. *Фактът* е фрейм, в който всички слотове съдържат терминални сто-нсти.
Ситуация е отделена по определени критерии група от фрейми. *Елементарна* е ситуацията, която съдържа само един фрейм.
Ако ситуацията съдържа само факти, тя се нарича фактуална и екстенсионална.
Глобалната ситуация съдържа всички фрейми от базата.
Изводът при фреймови модели използува действия като:
- преход от фрейм към фрейм,
- извличане на данни от фрейм,
- запълване на поле на фрейм,
- четене на слот,
- изпълнение на действия, указани в слот,
- …….
Изводът при фреймовите модели може да се раздели грубо на следните групи:
а) съпоставяне;
б) наследяване;
в) смесен.
А/ Целта на *съпоставянето* е по зададено описание на някакъв обект или ситуация да се намерят вече съществуващи в БЗ фрейми, които съответствуват в задоволителна степен на това описание. Резултатът от съпоставянето е някаква оценка на степента, в която конлретен фрейм съответства на зададеното описание. Най-често се

избира този фрейм, който има максимален брой съответни слотове, които съвпадат с данните за описвания обект или ситуация. Ако нито един фрейм не обхваща описанието на обекта или ситуацията, то се създава нов фрейм или най-близкият от съшествуващ фрейми се модифицира така, че да обхване и разглеждания обект или ситуация.

В/ Целта на *наследяването* е стойността на едно свойство да се запише и съхранява на колкото се може по-малко места. Освен свойства (описатели) могат да се наследяват и процедури (поведенчески х-ки на даден обект). Могат да се задават и различни ограничения върху наследяването.

Механизмът за *наследяване* се използува, когато ни интересува свързана с даден слот стойност. И тук, както и при семантичните мрежи, логическият извод се свежда до наследяване на свойства от класовете, към които принадлежи даденият обект/ клас. При това могат да се наследяват не само стойности, но и процедури - не само описателните, но и поведенческите характеристики на обектите. В зависимост от особеностите на предметната област и на разглежданите обекти от нея могат да се задават и различни ограничения върху наследяването на знанията от определени слотове и фасети.

Действува се в следната последователн:

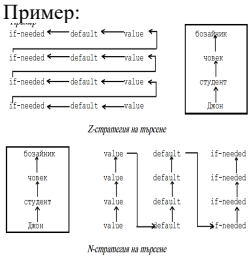
1. Тази стойност се търси във фасетите в следния ред: стойност, по подразбиране, при необходимост на текущо разглеждания слот.
2. Преминава се към точка 1 като за текущ фрейм се обявява фрейма, посочен в слота “от вида” (A kind of) на текущо разглеждания слот. Ако в А-К-О (A kind of) са записани имена на няколко фрейма предходници (тъй нареченото множествено наследяване), то се появява необходимост за управление на наследяването. В такъв случай най-често се използват следните две стратегии:

<u>А/ Търсене в ширина. (Z-търсене)</u>

Най-напред се разглеждат трите фасети (value, default, ifneeded)на разглеждания слот на дадения фрейм, след това се преминава към първия предходен на фрейма фрейм и докато се изчерпи списъка в А-К-О слота на този фрейм се повтаря търсене в трите фасети за същия слот. След това процедурата се повтаря на следващото ниво фрйми докато се изчерпи иерархията или се намери търсената стойност.

<u>Б/ Търсене в дълбочина. – N-търсене</u>

Най-напред се преглеждат фасетите value на дадения фрейм и неговите предходни докато се изчерпи иерархията и списъците от А-К-О слотовете на фреймите, след това се преглеждат фасетите default на фрейма и предходниците му и накрая фасетите if needed на фрейма и предходниците му. Пример:



Z-стратегия на търсене

N-стратегия на търсене

ПРЕДИМСТА И НЕДОСТАЪЦИ:

Основно предимство на този вид формализъм е естествеността на представянето, тъй като тосе доближава до това в човешкия мозък. Модулността, йерархичн структура при представянето им и възможността за задаване на стойности по подразбиране правят фреймовете предпочитан формализъм за ПИЗ. Чрез използването на фреймове за ПИЗ създаван на процедури, въвеждането на стойности по подразбиране и откриването на липсващ стойности се улеснява значително.

Отличителни предимст на фреймови модели – висока структурираност и възможността да бъдат вградени модулност и еластичност. Като основни предимства на фреймовите модели ще посочим следните:

- представянето е близко до начина, по който се съхраняват понятията в човешкия мозък;
- представянето предлага възможности за модулност и йерархия;
- съществува удобна възможност за задаване на свойства по премълчаване;

<u>Недостатъци</u>

Към основните недостатъци на този формализъм спадат неясната семантика, т.к. знанията, представени чрез фреймове, често могат да бъдат тълкувани по различен начин; недостатъчната изразителност на формализма в сравнение с предикатната логика от първи ред. Проблеми възникват и при управлението на наследяването на свойства припредстав чрез фреймове. Фреймовете са трудно програмируеми и възникват проблеми при извода на нови знания.

- липсата на формален механизъм като например при предикатн смятане;
– възможността разглежданите данни да могат да се тълкуват по различен начин. поради което е необходимо да се вземат мерки за еднозначно тълкуване;

Животно – (има своиство) диша

Животно – (е) птица – птицечовка, орел, пингвин.

- Всеки клас има определени свойства, които се наследяват.
- съществуват редица проблеми при наследяването, особено при множественото наследяване. При множествено наследяван от различните предходници даденият обект или клас могат да наследят противоречиви свойства. В такива случаи се постъпва по някой от следните начини:
- не се прави никакъв извод;
-предлагат се няколко (противоречащи си) заключения;
-въвеждат се тегла на различните класификаци и най-напред се работи с класификациите с най-голямо тегло.
-Трудно се изразяват също и *непълни знания*, които съдържат дизюнкция от свойства на различни обекти, като например: “Колата на Петър е зелена или колата на Иван е бяла.”.

<u>26.</u> Представяне на неточни знания. Вероятностни мрежи. Размита логика.

В повечето случаи експертите представят своите знания с различна степен на увереност. За изразяване на този факт се използув различни подходи.

Класификация на подходите:

А/ Числови – използват количествена мярка за несигурност. Число в интервала [0,1]. Те имат 3 под нюанса:

- о Вероятностен подход или случайността – най разпространен, който се опира на теория на вероятностите и матем. – несигурност по отношение на настъпването на 1 събитие.
- о Коефициент за достоверност
- о Размити величини и размита логика, базира се на теорията на възможностите или терория на размитете множества. Разликата е в това че случайността е несигурност по отношение на настъпването на 1 събитие а размитостта е несигурност по отнош на степента в която се е проявило 1 събитие.

Б/ Знакови – използват предимно качествена мярка за несигурност – логика по подразбиране, модални логики и т.н.

вероятността, която един субект съпоставя на едно събитие, винаги е свързана с неговите знания относно изпълнението на някакви предварителни условия. Така например, когато преди хвърлянето на зар казваме, че вероятността да се падне числото 3 е 1/6, това означава, че по-скоро казваме “ако са изпълнени (например) условията зарчето да е с идеално еднакви стени и при хвърлянето да няма никакви смущения и др., вероятността да се падне числото 3 е 1/6”. Ако се променят знанията на субекта относно изпълнен предварителни условия, вероятността, която той ще асоциира със събитието, също ще се промени.

Изречението “С вероятност 70% утре ще вали силен дъжд” съдържа несигурност и от случаен, и от размит характер. Знания, които съдържат несигурност от случаен (вероятностен) или размит тип ще наричаме съответно вероятностни или размити знания, а разсъжденията с такива знания – вероятностни или размити разсъждения.

ВЕРОЯТНОСТЕН ПОДХОД/РАЗСЪЖДЕН.

- Обективно и субективно схващане за вероятност

Според обективното схващане вероятността представлява отношението между броя на случаите, когато това събитие е вярно и общия брой случаи. Според субективното схващане вероятността се интерпретира като нечие вярване или събитие да е вярно. В интелигентните системи е по-подходящо *субективното* схващане, поради това, че са разсъждаващи обекти.

Това са няколко от методите и терм за представяне на несигурните знания чрез средствата на вероятносте:

• *Случайна променлива може да е булева T или F*. Това е терм в езика - величина за представяне на знания, която може да има няколко (вкл. безброй много) възможни стойности.

• *Област на променлива*: dom(x) = множеството от възможни стойности на x.

• *Твърдение:* булев израз от присвоявания на променливи (xi = vj). Например: (време = дъждовно) ∨ (болест = грип) ∨ ¬(температура = повишена).

• *Вероятност* = мярка за увереност в дадено твърдение (реално число между 0 и 1). P(A)=0 → 100% увереност, че твърдението A е лъжа; P(A)=1 → 100% увереност, че твърдението

• това е Безусловна вероятност – безусловна вяра (степен на убеденост на агента) че съждението A = T (е истина) при липса на каквато и било друга информация P(A)

•*Вероятностното* разпределение задава вероятността на всяка възможна стойност на променливата. Ако dom(x) = {v1,v2, …, vn}, то Sum n (i=1) P (X=Vi) = 1

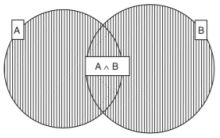
• *Априорна вероятност* – вероятност при отсъствие на каквато и да е информация.

• *Условна вероятност* – вероятност при наличието на информация за стойностите на други случайни променливи. Въвеждането на условности във вероятностите се осъществява с теоремата на Бейс (по долу). Условна вероятност каква е вероятността за случване на B, ако преди това се е случило A.

Например: P(температура = повишена | болест = грип). Тоест вероятността за повишена температ при болест грип. Вероятността съжденията A да е истина при условия че е истина съждение B. Означава се с P (A=T|B=T) съкратено P (A|B), където T е ст-ст на A или B която е с вероятност – 0,4 за 40% и т.н.

Ето и някои от основните зависимости (аксиоми) във вероятностите:

1во: вероятността, че или A е вярна, или B е вярна", се определя от следното правило: P(A∨B) = P(A) + P(B) – P(A∧B). вероятност на 1 дизюктивно съждение



2ро: незвисимост на 2 събития - ако 2 променливи A и B са независими (т.е. знанието на едното не променя вероятността на другото), когато P(A∧B) = P(A)P(B) => безусловна независимост.

Вероятността за случване на събитие B не зависи от това дали преди това се е случило или не събитие A. (След случване на A, B не се случва нито по-често, нито по-рядко.)

Хвърлени зарове, монети и т.н., последователно бъркане в различни съдове, от които се взема предмети - всеки следващ опит е независим.

3-то: A и B са несъвместими (т.е. никога не могат да се случат заедно), когато P(A∧B)=0

4мо: дефиниция за условна вероятност: Математическото определение за условна вероятност се записва по следния начин:

P(A|B) = P(B ∧ A) / P(B), P(B) >0.

⇨ Следователно: P(A∧B) = P(A|B) P(B).

Където P(B ∧ A) е *общата вероятност* двете събития да са се сбъднали, а P(B) е вероятността да се е сбъднало събитието "B" без оглед на другите обстоятелства.

Използва се за установяване на вероятност на твърдение A се отчита сбъдването на определено събитие B. Пр. на Бейс дава основание за разсъждение относно несигурността в много интелигентни системи. То изразява, че вероятността на събитие B, което зависи от сбъдването на A може да се изчисли ако са извършени безусловни вероятности на A и B и условната вероятност на A в зависимост от B.

P(B│A)=(P(A|B).P(B))/(P(A))

Ако A и B са независими събития, вероятност от комбинации или се изчисляват като произволна на вероятност, т.к. предположение за независимост на събитията не може да бъде обосновано за коректност на заключенията при подхода на Бейтс , се изисква изчисляване на пълните вероятности включително обединени им. В много случаи такова обширно набавяне на данни не е възможно или е много скъпо.

5-то: Правило Метод/Формула (теорема) на Бейс

Въвеждането на условности във вероятностите се осъществява с теоремата: P(A|B) = P(B | A)P(A) / P(B);

където P(A) — вероятност за настъпване на събитието A; P(A|B) — Условна вероятност за настъпване на събитието A при положение, че събитието B е настъпило (апостериорна вероятност); P(B|A) — Условна вероятност за настъпване на B при положение, че A е настъпило; P(B) — вероятност за настъпва на събитието B.

6-то: Условна независимост на A и B при дадено C (трета променлива: ако P(A|B∧C) = P(A|C) и P(B|A∧C) = P(B|C) при условие че съставното съждение B∧C е вярно!

ВЕРОЯТНОСТНИ МРЕЖИ (МРЕЖИ НА БЕЙС). МЕХАНИЗМИ ЗА ИЗВОД.

Подходи за извод.

А/ Чрез *съвместно разпределение*

Вероятностен модел на предметната област:

- Атомарно събитие: (X1 = V1) ∧ (x2 = v2) ∧ … ∧ (Xn = Vn), където xi са случайни променливи.

То описва конкретно състояние на предметната област.

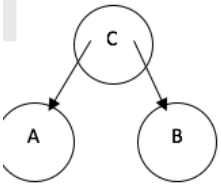
ОПРЕДЕЛЕНИЕ: *Съвместно вер. разпределен се нарича* n-мерна таблица с mi (i = 1,2,,.., n) с клетки по всяка размерност (ако xi има mi възможни стойности). Във всяка клетка се записва вероятността на съответното *атомарно събитие – всеки запис от таблицата.* Тъй като атомарните събития са несъвместими (т.е. взаимно изключващи се) и таблицата съдържа всички атомарни събития, то сумата от стойностите на всички клетки е 1. Възел без родители има само един ред в таблицата на априорните си вероятности,представящ вероятностите за всяка възможна стойност на променливата. Таблицата с всички възможни комбинации от ст-тите на булеви променливи се нарича *Съвместното вроятностно разпределение* (СВР). С нараств бр на променливите, бр на необходимите у-вни вер расте само линейно, докато размера на таблица на СРВ расте експоненциално.

Условната независимост м/у много променливи в сис-мата води до значителна икономия на редове в таблицата.

Б/ чрез изпозлване на формулата на Бейс. Мрежите на Бейс се създават от инжинера на знание и представляват *ориентирна графика за сбито (компактно) описание на зависимостите между променливите.*

Представяне на условните вероятности като стр-ра от данни – е вероятностна (Бейсова) мрежа. Във възлите на мрежата се задават даваните величини, връзките, свързващи двойките възли задават следствени връзки. Интуитивното значение на дъгата от възела X към възела Y е, че X оказва директно влияние върху Y. За всеки възел е дефинирана таблица за съвместно вероятно разпределение, което количествено определя ефекта от влиянието на „причинните" величини.

P(C):



P(A|C) P(B|C)
P(A|┐C) P(B|┐C)

Бейсовата (вероятностна) мрежа представлява насочен ацикличен граф, представен от

наредената двойка (V, E), където V е множеството от върхове (възли) на графа, а E е множеството от дъги, съединяващи върховете. Всеки възел на мрежата съдържа име на променлива. Дъгите от мрежата задават причинно-следствени връзки. Бейсовите мрежи се използват за представяне на зависимостите между тези променливи с цел компактно описание на съвместното им разпределение.
Интуитивното значение на дъгата от възела Хкъм възела Y е, че Y зависи от стойността на X или с други думи - X оказва директно влияние върху Y. За всеки възел е дефинирана таблица с условно вероятностно разпределение, която показва зависимостта на променливата от всички родители на този възел. Ако променливата няма предшественици, то за нея е дефинирана таблица с априорни вероятности.Всеки възел от мрежата съдържа таблица на условно вероятностно разпределение, която е дефинирана само върху неговите родители. Така вместо да се съхранява една голяма таблица на съвместно вероятностно разпределен за всички променливи, всеки възел съхранява своя малка таблица на условно вероятностно разпределен, която е дефинирана само върху възлите, от които той е зависим.

Обща процедура за изграждане на мрежите на Бейс в стъпки.

1. Избира се множество от подходящи применливи xi които описват предметната област
2. Избира се наредба за променливите. Поради начина по който работи процедурата за конструиране на межата на Бейс възлите оказващи директно влияние трябва да се включат в мрежата най-напред, т.к. те ще бъдат родители на всеки възел, върху който влияят. => коректният ред на добавяне на възли изисква най-напред да се поставят „причинителите" след това променливите върху които пряко влияят и така до „листата" възли които не оказват влияние върху други променливи.
3. Докато има ост. променливи се изп.
   - Избира се променлива xi и се добавя възел в мрежата за тази променлива
   - Установява се множеството родителски възли вече включено в мрежата, така че изискването за целева зависимост да е изпълнено.
   - Дефинира се таблицата с условните вероятности за xi

Така всеки възел е свързан само с предходните възли от мрежата => конструираната мрежа на Бейс е ацикличен граф.

ВИДОВЕ ИЗВОД В МРЕЖАТА НА БЕЙС (МБ). Съществуват четири вида изводи в Бейсовите мрежи. Това са диагностика, предсказване, междупричинен извод и смесен извод. При дадени стойности на част от променливите да се определи вероятността на стойностите на друго подмножество променливи
Това може да става по следните начини:
-от следствието към причината (диагностика)
-от причината към следствието (предсказване)
-междупричинен извод (м/у причините за дадено следствие)
Диагностиката,представл вероятността от следствието към причината. Пр. може да бъде вероятността P(Обир|Джон звъни). Предсказването е вероятността от причината към следствието, например P(Джон звъни|Обир). Междупричинният извод е вероятността между причините за дадено следствие. Пример за такъв извод е вероятността P(Обир|Земетресение). Смесеният извод е комбинация на предните три извода, например P(Аларма|Джон звъниΛ¬Земетресение).
МБ като правило са значително,по-компактно предоставяне на знанията за областта отколкото пълните таблици за съвместно разпределение на променливите.
Бейсовата мрежа осигурява пълно описание на домейна. Всеки ред в таблицата на пълното съвместно разпределение може да бъде изчислен чрез информацията в мрежата. Нека x1, x2, ..., xn са случайни променливи и P(v1,v2, ...,vn) е съвместната вероятност те да получат съответно стойности v1, v2, ..., vn. Тогава P(v1,v2,...,vn) = ∏n i=1 P (Vi |Parents (Xi)), където P (Vi |Parents (Xi)) е условната вероятност за xi = vi при условие, че са дадени стойностите на родителските променлив Parents(xi) на xi. Всеки множител на пълното съвместното разпределен е представен като умножение на подходящи елементи от таблицата на условната вероятност в Бейсовата мрежа. Таблицата на условната вероятност осигурява декомпозира представяне на съвместното разпредел.
Бейсовата мрежа е коректна репрезентация на домейна, само ако всеки възел е условно независим от неговите предшественици при съответната подредба на възлите в мрежата. Следователно, за да построим Бейсова мрежа с коректната структура на домейна, трябва да изберем родители за всеки възел, такива че да е изпълнено горепосоченото св-во. Родителите на възела Xi трябва да съдържат всички тези възли в X1,,., Xi-1, които директно влияят на Xi. Мрежата на Бейс често може да бъде далеч по-компактна отколкото пълното съвместно разпределение. Това свойство я прави предпочитана за справяне с домейни с много променливи.
Компактността на МБ е пример за едно от общите свойства на локално,структурираните системи. В една такава система всеки компонент взаимодейства директно само с ограничително множество от останалите компоненти. Вместо с всички останали компоненти, локалната структурираност обикновено се свързва с линейна, вместо с експоненциална сложност. В случай на МБ е обуслано да се предположи, че в повечето области всяка променлива се влияе директно от неповече от k други. Ако за простота има предвид булеви променливи тогава за представяне на таблица за всеки възел ще са необходими 2k числа => цялата мрежа може да се зададе с помощта на n*2k числа където n е броя на възлите. Докато при предст. на пълното съвместимо разпределен се изисква изпълнение на 2n числа. Например нека имаме 20 възела (n=20) и всеки от тях има най-много 5 родителя (k=5)
20*25 = 640 (МБ)
220 – пълно съвместимо разпределение
Основния недостатък на МБ е, че не може да се различи несигурността от незнанието.
Предимства на вероятностни мрежи са:
-дават възможност да се отрази, че твърденията се влияят взаимно.
Изискванията за изчисление на пълните вроятности
- Не може да каже какъв е резултата от кванторите и вероятностите
- Не е известно доколко вероятностите са подходящи за различни задачи
-Вероятността на събитята се определя субективно
Всяка БЗ основана на съждително смятане може се представи чрез вериятностна мрежа. Обратното не е вярно защото ВМ са много по прецизни.

<u>РАЗМИТА ЛОГИКА.</u>

Професор Заде създава теорията за размита логика (fuzzy logic) през 1973 г. Размитата логика (Fuzzy logic) е раздел на математическата логика, занимаващ се с теорията на неточно определената информация, която се изразява с приблизител стойности в интервал (напр. между 0 и 1) или с категории (напр., "топло", "горещо", "студено"). “Навън вали много слаб дъжд”. Това изречение съдържа неопределеност от типа размитост - какво означава “много слаб дъжд”, колко е “много е слаб” дъжда? То отразява степента на истинност, с която субектът характеризира проявата на събитието. Тази степен се представя като число в интервала [0,1] и за “слаб дъжд” тя може да е например 0.1, за “силен дъжд”- 0.8, за “много силен дъжд” - 1 и т.н. Формалните съждени, с които работят вероятн системи, не са принципно различни от формалните съждения в съждителното смятане - те са булеви променливи. От практичес гледна точка, това

ограничение е значително, защото сензорните показания на един агент (например, позиционни координати, данни за скорост, сила, температур, налягане и т.н.) обикновено не са булеви, а непрекъснати. За да може да се приложат системите, основани на правила в такива случаи, е необходимо да се намери начин разсъждението да се извършва върху реални променливи. Най-очевидната идея за решаване на въпроса, е да се въведат допълнителни булеви променливи, които дискретизират всяка реална променлива.

Пример: Ако променливата X приема стойности от интервала [-10;10], бихме могли да въведем трите булеви променливи PX, ZX, NX по следния начин:

PX=T(истина) ако X>=2, иначе PX=F(лъжа).

ZX=T(истина) ако 2>X>-2, иначе ZX=F(лъжа).

NX=T(истина) ако X<=-2, иначе NX=F(лъжа).

По този начин винаги точно една от тези променливи ще има стойност T(истина), а другите две ще имат стойност F(лъжа). Смисълът на тези три променливи, е както следва: PX="X е положителна", ZX="X е около нулата", NX="X е отрицателна". Нека е дадена и променливата Y, която на свой ред приема стойности от интервала [-5,5]. Аналогично въвеждаме булевите променливи PY, ZY и NY:

PY=T(истина) ако Y>=1, иначе PY=F(лъжа);

ZY=T(истина) ако -1<Y<1, иначе ZY=F(лъжа);

NY=T(истина) ако Y<=-1, иначе NY=F(лъжа);

Нека сега опитаме да изразим следната зависимост между променливите X и Y: "Колкото по-положител е променливата X, толкова по-отрицателна е променливата Y, и обратно". Това ще рече, че за отрицателни стойности на X, Y приема положителни стойности, и обратно -

за положителни стойности на X, Y приема отрицателни стойности; за стойности на X близки до нулата, стойностите на Y са също близки до нула. Тази зависимост може да се изрази чрез следните три правила:

1. NX →PY
2. ZX →ZY
3. PX →NY

Тук веднага обаче възникват два проблема:

1. Неточността (грешка) на разсъжденията чрез горните три правила. Нека, например, изведем стойността на Y за X=5: Понеже PX=T, ZX=F, NX=F, то по правило 3 NY=T, т.е. Y∈[-5,-1].

Този резултат не е особено информативен - полученият интервал е твърде голям, за да е полезен, и освен това явно в процеса на разсъждение е изгубена полезна информация.

Дискретизацията на реалната променлива X в само три булеви променливи е твърде груба и неточна. Ако не искаме да работим с интервали, можем например да използваме средната стойност за Y, в случая на (-3).

2. Невъзможността да се различи стойността на X=5 от X=9, например, защото едни и същи истинностни стойности ще бъдат присвоени на NX, ZX и PX и в двата случая (F, F, T, съответно). В резултат и в двата случая ще получим интервала [1,5] (или неговата средна стойност 3, ако предпочитаме да не работим с интервали).

РАЗМИТИ МНОЖЕСТВА – средства на такъв вид разсъждения можем да намерим в теорията на размитите множества и размитата логика.

Размито множество: двойката [S, m(S)], където S е обикновено множество, а

m: S →[0,1] е функция с дефиниционна област S, която дава резултат в интервала [0,1].

Функцията m(S) се нарича характеристична или функция на принадлежност на размитото множество.

Основни понятия и дефиниции:

1. *Размито множество.*

Нека X е реална променлива, а A1, A2,,.. An са размити променливи с техните функции на принадлежност mA1(X), mA2(X),,.., mAn(X), съответно. Можем да определим степента на истинност на всяка една от тези променливи за конкретната стойност на X. Степента на истинност на размита променлива A за конкретна стойност X ще означаваме с T(A), като

T(A)=mA(X).

Пр. Нека X е множество от обекти и A ∈X. Да си представим, че характеристичната функция mA (x) на множеството A може да приема произволни стойности в интервала [0, 1], тогава за елементите x, принадлежащи на множеството X, са възможни следните случаи:

а) x ∈A (mA (x) = 0) ;

б) x "принадлежи малко" на A (mA (x) ∈ [0, 1] и е близко до 0);

в) x "принадлежи доста" на A (mA (x) ∈ [0, 1] и не е близко нито до 0, нито до 1);

г) x "принадлежи твърде много" на A (mA (x) ∈ [0, 1] и е близко до 1).

Множеството A, представено по този начин, се нарича *размито множество A* над множеството X. Заде нарича характеристичната функция mA (x) функция на принадлежност на елемента x към множесвото A. Функцията на принадлежност като всяка една функция може да се опише аналитично, таблично, графично и т.н. Може да има и произволна форма в зависимост от това какво изразява. В практиката на размитите регулатори са се наложили линейно - отсечковите апроксимации на функциите на принадлежност и в този смисъл различаваме съответно триъгълни и трапецовидни форми. Това е така понеже, колкото повече стават степените на свобода на параметрите, толкова повече се усложнява задачата за оптимизиране (всъщност връзката е експоненциална).

На всяко размито множество можем да съпоставим съответна размита променлива. При тази уговорка можем да използваме еднакви имена на променливите и на размитите множества.

2. Операции върху размити множества. Тъй като обикновените множества могат да се разглеждат като частен случай на размитите множества, при които функцията на принадлежност приема стойност 0 и 1, то респ. операциите върху размитите множества могат да се разглеждат като разширение и допълнение на операциите върху обикновените множества. Трите най-често срещани, и напълно достатъчни за изграждането на размит регулатор, операции върху размити множества са, както и при обикновените множества - допълнение (NOT), обединение (OR) и сечение (AND).

ИЗВОДИ посредством размита логика:

Прилагането на размитата логика може условно да се раздели на три части:

-„Фъзифициране" на данните (fuzzification) - преминаване на данните във вид удобен за използване от размитата логика.

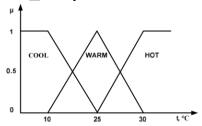
-Създаване на лингвистични правила отнасящи се до решавания проблем.

-„Дефъзифициране" (defuzzification) – обратно преобразуване на резултатите.

„Фъзифициране"

В първата фаза измерените реални входни данни се преобразуват по такъв начин, че да се представят като стойности, показващи принадлежността им към някоя функция на принадлежност.

При нашия пример за контрол на температурата, първо трябва да се създадат функциите на принадлежност на входната променлива air_temperature (отнасяща се до температурата на въздуха). Тези функции се дефинират като набор от стойности на променливата и тяхната принадлежност към функцията. При размитата логика е не само важно да се определи дали една входна променлива принадлежи на една функция на принадлежност, но и до каква степен и принадлежи. Една променлива може да принадлежи на различни функции на принадлежност в различна степен. На фиг. 3 са показани функции на принадлежност на променливата air_temperature.



Фиг.3

27. Планиране. Методи за планир Приложения.
Повтаря се с въпрос 8.
А. СЪЩНОСТ - Подход, при който се генер последователност от елем действия за изпълнение на задача или за постигане на определена цел, се нарича планиране.
Различията между планирането и търсенето се дължат на:
- Особеностите в представянето на състоянията, целите и действията.
- Спецификата във формирането и представянето на последователност от действия.
При търсенето всички действия се дефинират като оператор за преход. Оценъчната функция избира кой от наследниците е по-близо до целта, за да се приложат следващите оператори. Когато факторът на различ. расте и се прави сляпо търсене, изборът на подходящото състояние е сериозен проблем. Не може предварително да се отхвърлят част от състоянията и да се намалява времето за търсене.
Представянето на даден проблем като съвкупност от *състояния (как изглежда средата в момента)*, *действия (описание на средствата)* и *цели (как би трябвало да изглежда средата)*. В класич среди за планиране – *агентът* (planer) извършва крайни, позволени и конкретни дискретни елементарни (atomic action) действия. Кл момент е намирането на функция на добра евристика и как да се декомпозира и изпълни на части проблема в повечето случаи като дърво с подцели. Ген *план* представл линейно или частично нареда послед от *оператори* за вески който е описан *модел* на допуст действие. Разлика м/у търсене (действ се разглеждат като черна кутия, избор в резултат на евристич ф-ия => избора е проблем) и планиране (съст и целите се описват като изречения и има логически описание при което съст => някакво действие, позлва се метода на разбиване на задач на подзадачи)
Б/ Преставяне на съст, целите и действията – на език за класическо планиране език STRIPS.
STRIPS като класически език за планиране се състои от състоянията, целите и набора от действия:
Състоянието е съчетание от положителни буквали, които не могат да съдържат променливи и да привличат функции.
Целта, подобно на държавата, е съчетание от позитивни и земни (без променливи и без функции) буквали.
Действията (наричани още оператори) включват предпоставки и условия. И двете са представени като комбинация от литерали без функция. Предварителните условия описват състоянието на света, необходимо за извършване на действие, докато посткондикциите описват състоянието на света след извършването на действието.
Състоянията- чрез литерали които не съдържат функц. Имена (променливи) или Приема се, че светът е затворен - това означава, че всички условия, които не се срещат в състояние се считат за лъжа; Целите – може да съдържат променливи; Действията чрез оператори – индентификатор, предисловия, отпадащи литерали и добавяни литерали. Действие = Предусловие (положител литерали, посочващи какво трябва да бъде изпълнено в състоянието преди да може да се изпълнени действието) + Ефект (конюнкц на функционално-свободни литерали, описващи как се изменя състояни, когато действието бъде изпълнено. Оператора е приложим ако са изпълнени предусловията.
Наричаме схема на действие.
ПРИЛОЖЕНИЕ:
Задачата в “света на кубовете” – как да се реши зад за формиране на план на действията на ръката на робот.
Състоянията (начални, междинни, целеви) се представят чрез конюнкции от термове, в които се използват предикатите on, ontable, clear, holding, handempty.
Примери:
on(A,B) ∧ ontable(B) ∧ clear(A) ∧ holding(C)
ontable(A) ∧ on(B,A) ∧ handempty
on(x,A) ∧ on(y,x)
Моделиране на действията на робота
Елементарните действията на ръката на робота се представят с помощта на продукционни правила, всяко от които има три компонента:
• формула на предварит условие (Precondition);
• списък на изтриванията (Delete List);
• формула на добавянията (Add Formula).
Формулата на предварит условие е аналог на лявата страна на правилото при традиционните продукц. правила. Списъкът на изтриванията и формулата на добавянията са аналог на дясната страна на правилото при традиционните продукционни правила.
STRIPS in action, in the blocks world, which consists of a table, three blocks (a, b, and c) and a robot arm that can move blocks around. two predicates to describe the world: *On (x, y)* means that block x is on top of block y & *Clear (x)* means that block x has no block on top of it. On (a, t) means that block a is on the table. *Clear (t)* will always be true because we assume that the table is large enough to hold at least three blocks at once. Our goal is to place block c on top of block a, which can be stated as On (c, a). start state can On (a, t), On (b, c), On (c, t), Clear (b), Clear (a), Clear (t). Operator schema: MoveOnto (x,y), which means “move object x from wherever it is, and place it on top of object y.”

В/ ОСН. МЕТОДИ за Т в ПС – или т.н. планирането на алгоритми. ПОДХОД свързан с езика STRIPS и ИЗПОЛЗВА ТЪРСЕНЕ В ПРОСТР ОТ СЪСТОЯНИ И простр на ПЛАНОВЕТЕ. Чрез насочен граф, възлите в който преставл състоянията а дъгите – операторит в STRIPS. В последните години стана ясно, че STRIPS не е достатъчно изразителен за някои реални ситуации. В резултат на това бяха разработени много различни варианти на езици.
Тъй като описанията на действията определят както предусловията, така и ефектите, е възможно търсенето да се осъществи в една от двете посоки:
- *Планиране чрез прогресия на Действието* -търсене напред от началното състояние. Разглежда ефекта от всички възмож действия в дадено състояние. И кое трябва да се приложи. Има голяма възможност да се сбърка при висок фактор разклонения защото има голямо простр на търсенето. Добрата евристика е задължител за постигането на ефективно търсене. За начално състояние при търсенето ще считаме началното състояние на проблема. Като цяло, всяко състояние е множество от положите основни литерали. Литерали, които не се споменават са лъжа. Действията, които са приложими в състояние, са всички онези чиито предусловия са изпълнен – на принципа: добавяне на положителни ефекти и изтриване на отрицател.
-*Планиране чрез регресия на Целта* - търсене назад от целта. За достигане на целта, това което трябва да бъде вярно е предишното състояние. Каква е следващата подцел да се удовлетвори. Предпочит метод. Но как да определим предшествес и кои са състоянията, които след прилагане на някакво действие биха водили до целта? В допълнение към изискването на действията да достигнат до някакъв желан литерал, трябва да се уверим, че тези действия не премахват всички желани литерали. Действ, което удовлет тези ограничен, се нарича *консистентно или последователно*. Главното предимство на този алгоритъм е, че предприетите действия са винаги уместни. Често факторът на разклонение е много по-малък, отколкото при алгоритъм за планиране чрез прогресия. Основната идея е да се изчисли колко действия биха били необходими до достигането на целта – правилната евристика нужна за всеки от посоч по горе 2 метода. Търсенето може да се осъществи чрез всеки стандартен алгоритъм за търсене. Прекратяваме процеса, когато е генерирано описание на предшественик, което се удовлетворява от начален състояние на проблема.
Др. подход е търсене в простр на плановете (GraphPlan) – планиране,

като се започне от първоначалното
състояние и се работи към целта чрез извличане плановете. Графиката на планиране се състои от няколко нива. Първото ниво (на нулево) съдържа предложенията
които са верни в началното състояние на проблема. Следващото ниво на графиката съдържа действията, които могат да бъдат извършени в това състояние. Нивото
след това съдържа държавите, които могат да бъдат доведени чрез извършването на тези действия. Следователно всяко равномерно ниво в плана представлява състояние и всеки нечетно число представл действия. Крайното състояние в графиката представлява целта. Algoritym: Първо, предложенията, които описват целта, се сравняват с сегашно състояние. Ако всички тези предложения са налице и няма два от тях
са свързани с mutex връзка, тогава е възможно, че решение вече е е достигнато. На този етап се изпълнява втора фаза на алгоритъма, за да се опита извлечете план от настоящия графичен план. Ако текущото състояние не съдържа всички необходими предложения, тогава
следващото ниво на графиката за планиране се получава чрез прилагане на всички приложими
операторите и да определят всички възможни предложения, които могат да бъдат изпълнени от тези оператори. Този алгоритъм се повтаря, докато не бъде намерен подходящ план или докато не може да бъде
че няма план.

- *Частично наредени планове.* Досега разгледаните планирани (прогресивно и регресив) са видове пълно наредени планирания. Можем да изградим пълно нареден план, работейки с частично наредени планове, Планът, който изграждаме, включва две независими последоват от действия примерно обуването на чорап за ляв и десен крак. За разлика от пълните наредби, тук не заявяваме строг ред на изпълнение на тези последователности.

Планирането започва с непълен частичен план. 1 частичен може да се разглежда като множеств от пълни, всеки от който изпълня множество от ограниче. И се използва 2 вида оператори – прецезиран и модифиц. Всеки алгоритъм за планиране, в който могат да се поставят две действия в план, без да се уточнява кое се случва първо, се нарича *алгоритъм за частично наредени планове*. Реше е представено във вид на графика на действията, а не като таката тяхна последоват. Състоянията са, най-често незавърше, планове. Празният план съдържа единствено начал и финалното действие. Всеки план се състои от 4 компоненти:
*Множество от действия – това са стъпките на плана.
*Множество от наредени ограничения: A<B. Циклите представляват противоречия.
*Множество от причинно-следствени връзки: A--p-->B
*Множество от отворени предусловия – ако предусловието не е достигнато от някое действ. на плана.

С частично нареденият план действаме по следния начин:
1. Избираме едно отворено предусловие.
2. Търсим всички консистентни планове-наследници, т.е. поредици от действия,
които удовлетворяват условието.
3. Добавяме причинно-следствените връзки и ограниченията към плана (A--p-->B
и A < B).
4. Разрешава конфликтите между новата причинно-следствена връзка и
съществуващите
планове, както и между новото състояние A и съществуващите планов.

Един план е консистентен, ако в него няма цикли в наредените ограничения, и няма конфликти в причинно-следствените връзки. В процеса на действие, частично нареденият план се преобразува в пълно нареден. Консистентен план, който не съдържа отворени предусловия е решение. Вече сме готови да формулираме задачата, която частично наредените планове решават. Ще започнем с формулиран на подходящо твърдение за проблемите на планиран. Както обикновено, дефиницията включва началното състояние, действията и целта.

Началният план:
*съдържа състоянията Начало и Край
*ограничението Начало < Край
*не съдържа причинно-следствени връзки
*всички предусловия в Край са отворени

Функция на наследника – избира произволно едно отворено предусловие p на действие B и генерира план за наследника на всяко възможно съответствие на действието A, което
удовлетворява p.
=> Тестването на целта проверява дали планът е решение на първоначално
поставения проблем,
т.к. са генерирани само консистентни планове, тестването трябва да провери дали не сесъдържа отворени предпоставки.

28. Експертни системи - характерни особености. Типични задачи за решаване с експертни системи. Основни области за приложение на експертни системи.

ЕС е комп. програма която извършва експертиза с нивото на експерт. Представят се знания и се извършва логически извод на базата на налични (експертни) знания от дадена предметна област с цел решаване на типични задачи от тази област или получаване на подходящи съвети. При ЕС трад подх осн на съотношение:
АЛГОРИТМИ + ДАННИ = ПРОГРАМА е заменя от подход на който в осн стоят знания и логически изводи: ЗНАНИЯ + ЛОГ ИЗВОД = СИСТЕМА. ЕС като правило решават реални задачи и често са предмет на търговски интерес; към ЕС се поставя изискване за достатъчно бързо и точно получаване на търсените решения (поне толкова бързо, колкото те се получават от човека експерт, и поне толкова точни, т. е. правилни решения, колкото точно би ги взел експертът). ЕС трябва да могат да обясняват и обосновават взетите от тях решения.

Често терминът системи, основани на знания (СБЗ, Knowledge Based Systems, KBS) се използва като синоним на термина ЕС. В действителност първият термин е по-общ, т. е. всяка ЕС е СБЗ, но обратното не винаги е вярно. Система, базирана на знания, е всяка система, в която се решават задачи, като се използва знаково (текстово) представяне на знания. Така например програма, способна да води диалог за времето, вероятно ще бъде СБЗ, дори и да не може да прави експертни метеорологич прогнози, но ЕС в областта на метеороло- гията трябва да може да прави това.

Следователно в ЕС са включени експертни знания от дадена предметна област, които се използват за решаване на специфични задачи от тази област. Процесът на създаване на ЕС често се нарича инженерия на знанията.

Тя се харктеризира със следното:
1-во Работи със знания – от формални теореми до набор от евристики и обобщен опит на експерти.
2-ро Нейната гъвкавост – в процеса на своята работа ЕС обновява своята база от знания и изключва остарели знания.
3-то ЕС е така устроена че тя решава конкретния проблем по една обща “методика”, която се нарича механизъм за извод. ЕС може да представя и да разсъждава със знания в някаква област; областта е тясно определена решава конкретни проблеми (задачи) или да дава съвет в същата област.
4-то ЕС е разбираема дружелюбна към потребителите отговаря на техни въпроси и по този начин обучава самият потребител.

Експертната система разчита на натрупана база знания в определена област, въз основата на която с машина за разсъждения (inference engine) "разсъждава" върху факти в същата експертна област.

Най-важно у-вие за разработване на ЕС е да има експер, който да може да се справи с проблема! Не може да се създаде ЕС за нерешима досега проблем. Др. условие е обхватът на проблемите с които се занимава ЕС. Трябва да бъде достатъчно стестнен за да се получи ефективна база от знания и да се да има достатъчно вирок кръг от потребители. Трябва предварително да се оцени дали има алгоритмично

решение на проблема. Ако то съществува не е нужно да се изгражд ЕС.
ЕС се ползват:
1 Ако вашият проблем е толкова сложен, че алгоретмично решение не е известно.
2 Решението на проблема да е от компетенцията на високи специалисти/ експерти
3 Икономически и технически е изгодно да се репродуцират знания чрез компютри.
ОБЛАСТИ НА ПРИЛОЖЕНИЕ-
ЕС се използват за реш на широк кр задачи:
Обикновено ЕС се използват за решаване на следните типове задачи:
• Интерпретация на данни (например звукови сигнали);
• Диагностика на повреди или заболявания;
• Структурен анализ на сложни обекти (например химични съединения);
• Конфигурация на сложни обекти (например компютърни системи);
• Планиране на последователности от действия.
Статистика на използване на ЕС:
1-во място – Военни и космически технологии. Математика.
2-ро място – Икономика и управление на крупни фирми и в производството.
3-то място – Диагностика (медицината) - MYCIN е ЕС в областта на медицинската диагностика, Химия - DENDRAL е ЕС за химически анализ на възможните конфигурации на множества от атоми, Геология - ROSPECTOR е ЕС в областта на геологията
4-то място – Експериментални

29. Структура на експертните системи. Класификация на експертните системи.

СТРУКТУРА:
ЕС използват знания, представени във вид на продукционни правила. Такива ЕС обикновено се наричат ЕС, основани на правила. Съответният термин на английски език е Rule-Based Expert Systems.
За да може да функциория ЕС, нейни компоненти включват БЛОКОВЕ:
-потребителски интерфейс (UI) - Интерфейс с потребителя – анализира изреченията на естествен език от потребителя към системата и обратно;
-блок-подсистема за обяснение (представя "разсъжденията" на потребителя) - Най-прост вариант за реализиране на обяснението – системата запомня състоянието на базата данни и правилата, които са били използвани от началото до края.;
-база знания (правила);
работна памет (натрупани факти– знания за предметната област; знания относно методите за решаване на задачи; знания на езика на който се води диалог; подсистема за диалог на естествен език метазнанията.;
- машина/блок за извод (интерпретатор на правила) и общо управление на ЕС) разсъждения (inference engine),
-подсистема за натрупване (извличане) на знания в базата (натрупване, което не е точно обучение - може да е простовъвеждане/ добавяне от експерта).
Общата структурно-функционална СХЕМА:

Спомагателни блокове: блок за обяснения, двата редактора, анализ на базата данни.
*Анализ на базата данни – той трябва да потърси правила за да изведе в системата (формула) и да обогатява базата от знания.
*Редактор на БД – дава възможност за нови данни и механизъм с който да увеличим БД
*Редактор на БЗ – възможност за включване на нови правила
При ЕС често се поставя едно важно изискване към БЗ и интерпретатора - наличие на средства за представяне и използване на непълни и/или неточни (несигурни) знания и данни.
ВИДОВЕ И КЛАСИФИКАЦИЯ:
Обособят на 2 гр: *анализиращи и синтезиращи* ЕС
Типични задачи решавани с помощта на ЕС са *анализиращите* където се използват логически изводи за интерпретация и извеждане на закл от натрупани данни:
1.*Следящи* – в реално време да открият по поведението на системата опреде ситуации.
Класификации – разделяне и подреждане на обектите в съответствие с техни признаци. При следенето на състояние на пациенти, атомни реактори или др.
2.Интерпретиращи. Използват данните от датчици за описване на реална ситуация, работтят с реални данни, които често санепълни – Системи за разпозн на образи. Анализ на данните и предаването им в определен смисъл.
3.Диагностициращи – откриват вер причини за нарушения в функц х-ки на 1 система. Като се пази база на описаната ситуация и на знания за влиянието на отделни неизправности. В областта на медицината, при реш иженерни проблеми. (да може да се определи типа на дадено заболяване или др. след като разполагаме с опр. брой измервания на клиента/обекта)
Синтезиращи:
Използват класове от обекти от по-ниско ниво за формиране на обекти по-високо ниво:
1.Прогнозиращи - Извеждат по логическ път предполагаемите следстви от дадена ситуация.
Прогнозирания на времето, търговски проучвания на пазара. Предвиждане на възможни последствия от определен тип ситуации.
2.Планиращи – построяване на последователност от действия за постигане на опр. цел. В военната област, планиране на маршрути.
*3.*Проектиращи - конструират нови обекти или конфигурации от обекти на осн на набор от ограничения. Тъй като то е близко до планирането много от тези системи съдържат механизми за създване и корегиране на план за проектиранеА)

Универсални ЕС – програмни среди за разработка на ЕС
Б) Специализирани ЕС – ЕС за конкретната област

30. Интерфейси в експертните системи. Методи за придобиване на знания. Обяснения в експертните системи

4.1. Механизми извод.
Служи за избор и прилагане на подходящи знания към базата данни. Служи също за обосновка на решението, автоматично натрупване на нови знания и натрупване на служебна информация. Зависи от модела за представяне на знания.
Третият модул е за обмисляне на решението. Прилагат се неточни и приближени методи заедно със достоверните възниква необходимост от убеждаване на потребителя за достоверността на решението. Но се преследват и други цели, като
- обучение на потребителя
- определяне недостига от знания
- локализиране на грешки в системата
- две възможни запитвания обикновено се осигуряват - защо, как?
Логическия извод в права посока, обикновено се представя като дърво на извода. В процеса на консултация, експертната система строи дървото на извода и го запазва в паметта са си във вътрешна форма. Когато се пита защо е определен въпрос, системата проследява дървото на извода нагоре и се виждат по-високите цели които тя се опитва да постигне. А когато ни е зададен въпроса "Как", се проследява въпроса надолу, за да се види кои под-цели са удовлетворени, за постигане на крайната цел.
Системата за обяснение представлява обхождане на дървото на извода, като изпълнява търсене в него.

31. Проектиране на експертни системи - участници, етапи. Основни задачи на инженера по знания.

Съображения за изграждане на експертни системи:
• Може ли проблемът да бъде разрешен ефективно чрез конвенционално програмиране?
• Има ли нужда и желание за експертна система?
• Има ли поне един експерт в областта на

човешките ресурси, който е готов да сътрудничи?
• Може ли експертът да обясни знанията, които инженерът на знанието може да разбере.
• Знанията за решаване на проблеми са предимно евристични и несигурни?
При разработката на ЕС се използва концепцията на бързия прототип. Същността на този подход е в това, че при разработката крайният продукт не се създава веднага. В началният етап се изгражда прототип на ЕС, който трябва да осигури решаване на типични задачи от конкретното приложение при минимално време и трудоемкост. За целта се използват различни инструментални средства, които ускоряват процеса на проектиране. В случай на успех, специалистът по знания, заедно с експерта разширява знанията на прототипа за проблемната област. При неуспех може да се разработи нов прототип или да се стигне до непригодност на използваните методи за даденото приложение. Преобразуването на прототипа в краен продукт обикновено довежда до препрограмиране на ЕС на език от ниско равнище. Трудоемкостта и времето за създаване на ЕС в голяма степен зависят от типа на използваните инструментални средства.
УЧАСТНИЦИ:
Основните типове потребители на ЕС, които работят с ЕС на различни етапи от процеса на създаване и експлоатация на системата и основните типове дейности, които те извършват:
-спец. по създване на инструментални средства – изгражда ядрото на ЕС чрез средства за извличане в диалогов режим на знанията на човека-експерт и попълване на БЗ и ЕС.;
- експерт в предметната област - ;
- обслужващ персонал – добавя данни;
- краен потребител;
- информатик по знания – интервюира експертите в пред област и завно с спец по създване на инстр средства използва ядрото на ЕС.
- празни ЕС (ядра на ЕС) – по същество вкл формализъм за представяне ан знанията и реализацията на съответните средства за извод (интерпрататори). Формализъм за представяне на знания – чето фрейми и правила, както и средства за орг на диалога с потребителя и експерта.

ЕТАПИ
В хода на работата по създаването на ЕС се налага определена технология на разработка, включваща следните етапи: идентификация, концептуализация, формализация, изпълнение, тестване и опитна експлоатация:
1. Дефинирането на проблема идентификация включва: разбиране на проблема; идентифициран на характеристиките на проблема; определяне целите на процеса и точно дефиниране на методологията, която е необходима за решаването на проблема. От тези дейности, идентификация на проблема и неговият тип и обхват създават много проблеми на разработчиците. Често проблемът, разгледан като много обширен и сложен, трябва да се намали до обозрими размери, чрез фокусиране на малки, но интересни подпроблеми и изпълняване на стратегии за решаването им.
2. Събиране, представяне и координация на знанията концептуализац - включва добиване на знания; представяне на знания; проектиране на механизма на извод; избор на програмни средства и координация на знанията.
Знанията за проблемната област се добиват от различни източници: от изучаване на публикуваната литература, от експерти в дадената област или от наблюдения и анализ на архивни данни. Базата знания съдържа различни категории знания:
- знания за обекта – факти, които дават описание на реални ситуации, релевантни на проблема;
- знания за ситуациите – необходими налични знания за ситуациите, които са се случили или се случват. Необходимо е да се създаде формализъм за описание на такива знания;
- знания за изпълненията – това са знания за прилагане на умения как да се ръководят процесите. Трудно е да се прекара граница между знанията за обектите и знанията за изпълненията.
3. Формализацията - включва изразяване на ключовите концепции и релации по някакъв формален начин, обикновено с фрейми или някакъв програмен език или подходящо средство за разработка на експертни системи. Евристично програмиране – базира се на използване на придобит опит при създаване на програмата и рационални идеи за решаване на задачата. Прилага се или когато има точен математически апарат (метод), който не води до ефективно решение, или когато липсва такъв метод. Подходът не винаги води до оптимално решение, но целта е получаване на възможно най-близкото до него.
4. Изпълнението - включва превръщане на формализираните знания и процедури в работеща компютърна програма. Много експертни системи се изграждат в началото като програма-прототип. Конструирането на програмата изисква съдържание, форма и интеграция.
Съдържанието идва от знанията за проблема, направени пълни и структуриране чрез формализацията, т.е. структури, данни, правила за извод и управляващи стратегии, необходими за решаване на проблема. Формата е специфицирана чрез езика, избран за разработване на системата.
5. Тестването - включва оценяване на изпълнението и полезността на програмата-прототип и ревизиране дали е необходима. Експерт по проблема оценява прототипа и помага да се преработи. Веднага щом прототипа работи върху няколко примера, той може да бъде тестван върху много проблеми, за да се оцени изпълнението му. Възникналите проблеми трябва да се изчистят от разработчиците. Това оценяване разглежда дали системата прави заключения, с които експертите са съгласни, дали правилата за извод са коректни и пълни. Експертната система трябва да бъде тествана и изчистена в лабораторна среда преди да бъде представена за тестване от потребителите.
Много експертни системи започват като демонстрационни прототипи, т.е. малки демонстрационни програми, които работят с част от проблема, за който евентуално са предназначени. Те се използват за да покажат как се прилага технологията на експертните системи и за да се тестват основните концепции на системата.
6. Опитна експлоатация - проверява пригодността на ЕС за крайните потребители. В зависимост от резултатите може да се стигне до осъществяване на модификация на ЕС.
ОСНОВНИ ЗАДАЧИ НА ИНЖ ПО ЗНАНИЯ:
Терминът “придобиване на знания” най-често се използва за означаване на процеса на използване на специални инструментал програмни средства, предназначени за пряко общуване с експерта. Придобиване на знания от една ЕС е свързан с поредица от интервюта между *специалист по знанията* и експерт(и) от предметната област. При това общуване действително се придобиват вече готови фрагменти от знания в съответствие със структури, предварително определени от инженера на знанията и “заложени” в съответната програмна система. Повечето от тези инструментални средства са специално ориентирани към конкретни ЕС, предназначени за работа в тесни предметни области с предварително определени модели на представянето на знанията, т.е. като правило те не са универсални.
Въведените по долу класификации на методите и предметните области могат да помогнат на инженера на знанията при първоначалния избор на метод или група методи за извличане на знания. Възможно е обаче по време на конкретната работа да се окаже, че първоначалните преценки са частично или дори

напълно погрешни и в такъв смисъл да се наложи преоценка и промяна на избраните методи.

Класификация на методите за практическо извличане на знания, в която класификационен признак е използваният източник на знанията.

Комуникативните методи за извличане на знания обхващат методи и процедури за контакти на инженера на знанията с непосредствения източник на знанията – експерта, а *текстологичните методи* за извличане на знания включват методи за извличане на знания от документи (методики, пособия, ръководства) и специализирана литература (статии, монографии, учебници).

Обикновено инженерът на знанията комбинира различни типове методи, например отначало изучава литература и след това беседва с експертите или обратно.

Комуникативните методи могат да бъдат разделени на две групи: активни и пасивни. Пасивните методи предполагат, че водещата рола в процеса на извличане на знания се играе по някакъв начин от експерта, а инженерът на знанията само протоколира разсъждения на експерта по време на реалната му работа по вземането на решения или записва това, което експертът счита за необходимо да разкаже по времето на отделни лекции. Обратно, при активните методи инициативата напълно е в ръцете на инженера на знанията, който активно контактува с експерта по различни начини – чрез игри, диалози, беседи и т.н.

Както беше посочено, процесът на придобиване на знания с помощта на интервюта между специалиста по знанията и експерта от предметната област е много бавен и неефективен. Той се смята за "тясното място" на процеса на създаване на ЕС. Най-важните причини за тази неефективност са следните:

- обикновено специалистите (експертите) в дадена предметна област трудно формулират знанията си пред неспециалисти (какъвто е специалиста по знанията), тъй като неспециалистите не познават и не разбират добре съответната терминология и съответния жаргон. Почти във всички случаи за експерта се оказва трудно (дори с помощта на специалиста по знанията) да формулира знанията си в такива термини, които са ясни, точни, пълни и непротиворечиви, т.е. използваеми в ПС;
- т.нар. стратегически данни (които обосновават методите за решаване на задачи и вземане на решения) в много предметни области трудно могат да бъдат формулирани в термините на някаква математическа теория или детерминиран модел, чиито свойства са ясни и разбираеми и могат за бъдат точно определени. Но наличието на този тип знания е много важно както за ефективността на системата, така и за възможността за генериране на задоволителни обяснения на взетите решения;
- експертите знаят не само факти и свойства на явленията от предметната област Голяма част от най-ценните им знания, например знанията, свързани със собствен опит как дадена тежка задача е най-добре да се раздели на подзадачи, трудно могат да бъдат извлечени и формулирани за използване от други хора.

Всички тези проблеми, свързани с интервюирането на експертите от други лица, са причина за създаването на различни методи и средства за автоматизация на процеса на придобиване на знания. Тази автоматизация обикновено се търси в две направления:

- създаване на програмни средства за автоматично извличане на знания, при които знанията на ЕС се придобиват в процес на диалог между ПС (обикновено наричана редактор на знания) и човек-експерт.

Следователно по този начин се избягва участието или се намалява ролята на специалиста по знанията;

- създаване на програмни средства за машинно обучение и самообучение (Machine Learning) - системата може да се обучава на базата на примери, аналогии, анализ на експериментални данни и др. Следователно така се намалява ролята на експерта.
- Евристично програмиране – базира се на използване на придобит опит при създаване на програмата и рационални идеи за решаване на задачата.

32. Анализ и синтез на говор. Речеви интерфейси.

Текстовите интерфейси са "традиционни", в известен смисъл на думата. Много езици за програмиране използват оператори, които са базирани на естествен език. Естествените езикови интерфейси (ЕЕИ) представляват тип от Компютърния човешки интерфейс, където лингвистиката и нейните принадлежащи фрази, глаголи и условни случаи служат за потребителски контроли за създаване, селектиране и модифициране на информацията в различните приложения По подобен начин традиционните търсачки могат да бъдат описани като "прост" Естествен езиков интерфейс.

Формален език – ез за програмиране.

Естествения език – който се състои от думи който има няколко форми (не само лексикална – неделими по смисъл думи) а и смисленето съчетание от лексими примерно.

Същност на понятието "ограничен естествен език" (ОЕЕ). Като правило под общуване с компютрите (с програмни системи) на естествен език се разбира общуване на т. нар. ограничен естествен език (псевдоестествен език; език, близък до естествения).

Ограничаването на естествените езици се извършва най-често в следните два аспекта:

- ограничаване на структурата на допустимите фрази (опростяване на синтаксиса на езика);
- ограничаване на предметната област (създаване на езици за общуване в дадена тясна предметна област – така се ограничава речникът на системата и се ограничават възможните интерпретации на отделните фрази, например се избягва опасността от омонимия).

В обхвата на ИИ същността на *общуването - речевия интерфейс* включва процес, при който един агент (говорещ) се стреми да направи така, че друг агент (слушащ) да промени своите знания и/или цели. => Общуването на ОЕЕ има два аспекта (две страни):

-анализ на (фрази на) ОЕЕ, т. е. възможност за разбиране на команди, заявки и т. н., формулирани на този език;

-синтез (генерация) на (фрази на) ОЕЕ, т. е. възможност за отговор на ОЕЕ.

*Етапи на общуването за **говорещия***:

• намерение – вземане на решение относно информацията, която ще бъде включена в съответното съобщение.

• пораждане – превръщане на подбраната информация в съобщение, т.е. низ от думи;

• **синтез** – извеждане на съобщението във вид, определен от условията на средата, в която се извършва общуването.

*Етапи на общуването за **слушащия***:

• възприятие – превръщане на съобщението в низ от думи (разпознаване на символи, на говорима реч и т.н.);

• **<u>анализ</u> (NLP natural language processing) – установяване на информационното съдържание на съобщението на ниво:**

-синтаксис: определяне на функционалната структура на съобщението. На това ниво на анализ обикновено се извършва групиране на думите в йерархични структури, наречени фрази;

-семантика: установяван на значението на отделните думи и фрази;

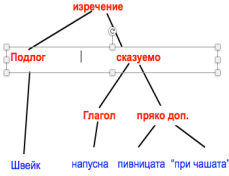
-прагматика: поставяне на значението в контекста на общуването (извличане на възможните "полезни" изводи от значението в контекста на знанията на слушащия или особеностите на ситуацията);

• усвояване – евентуално модифициране на базата от знания или целите в зависимост от отношенията между агентите.

Основ проблем пред който е изправен ИИ при обработката на ЕЕ е *многозначността* му.

Характерно свойство на естественоезиковото

общуване.
Отстраняването на многозначността е един от основните проблеми на обработката на естествен език. Нива на многозначността:
• фонетично (“Петко рита” и “пет корита”);
•синтактично (“Ям макарони със сирене” и “Ям макарони с вилица”: предложната фраза “със сирене” пояснява съществителното “макарони”, докато предложната фраза “с вилица” пояснява глагола “ям”);
• семантично (омоними);
• прагматично (“Знаете ли колко е часът?” – буквален въпрос относно знанието, с което разполага слушащият, в контекст обикновено еквивалентен на “Колко е часът?” или “Смятам, че вече е твърде късно.”).
Задачите ПО ОБРАБОТКА/АНАЛИЗ НА РЕЧЕВИ СИГНАЛИ
Речта притежава характерни признаци, определяни от говорещия. Следователно анализът на речта ще има за цел да открие - Кой говори?, а оттук и да разпознае говорещия (проверка и идентификация).
Компютърът идентифиц и проверява говорещия чрез акустичен отпечатък или предварителна проба.
Друга основна задача на анализа на речта е да анализира - Какво е казано?, т.е. да се разпознае и разбере съдържанието на говорния сигнал.
На базата на говорната последователност се генерира съответен текст.
Следваща задача при анализа на речта се отнася до това - Как? е казано дадено изречение или дума. Така например, изречението звучи по различен начин в зависимост от емоционалното състояние на говорещия.
=> Последователни стъпки нужни при анализа на естествения език в посочена по долуСХЕМА: изречения (последователност от думи) -> морфологичен анализ (анализатор) -> синтактичен анализатор (думи с характеристики /род, число, корен/) -> сематичен анализатор (изречение с характеристики /разбор/) -> знания във вид удобен за ЕС (фрейми, мрежа, предикати)
В посока изречения-знания – анализатор
В посока знания-изречения – синтезатор
Оформят се следните основни етапи на обработката и анализа на реч и фрази на ОЕЕ с различни анализатори:
1/ Акустичен анализ - има за задача да преобразува звука в последователност от цифри, които дават честотата, амплитудата. Или това е дигитализиране на сигнала
2/ Фонетичен - преди морфологичния анализ може да се наложи да се обработи сигнала фонологически – да се индентифицират думите от общия фон и др. звуци. В получената последователност от цифри при 1/ се откриват градивните частици на речта (фонеми). Езиковите елементи, обработвани като сигнали, се наричат фонеми. Или различаване на сегменти на думата, които могат да бъдат асемблирани в цели думи.
3/Лексичен /морфологич/ анализ: трябва да се получи от чисти разпознати фонеми, да ги преобразува в букви, те се свързват една в друга и се получават думи. Обособяват се на отделните думи (лексеми) от фразата. Морфемата е най-малката значеща единица в езика. Морфологичният анализ означава определяне на думата като част на речта и на всички нейни граматически характ-ки. Резултатът е линейна структура (обикновено списък) с елем – последователните лексеми от анализираната фраза.
4/Синтактичен анализ: от линейно представяне към формиране на структурно описание на анализираната фраза, най-често под вид на синтактично дърво, което дава представа за стр-ра на фразата като послед.



От синтакт единици (гупи), в съотв. С опис на синтаксиса на езика. Синтаксисът е дял от граматиката, който се занимава със строежа на изречението и с неговите части. В тази стъпка на гласовото разпознаване изреченията, които са линейни последователности от думи, се преобразуват в структури, които показват как думите се отнасят една към друга. Тази стъпка на граматически разбор, наречена парсване (*parsing*), превръща линейния списък от думи в изречението в структура, която дефинира представените единици от този списък.
5/ Прозодичен анализ – извлича се интонацията, ударенията, паузите
6/*Семантичен анализ:* Вкл заместване на част от поддърветата на синтактичното дърво с имена на конкретни обекти от сцената (предметната област) – редукция на синтактичното дърво. Семантиката е дял от граматиката за значението на думите, на техните части и съчетания. При семантичния анализ на структурите, създадени от синтактичния анализатор, се присвояват стойности. В повечето езикови пространства словосъчет “Безцветната зелена жаба” ще бъде отхвърлена като семантично неправилна. Тази стъпка съпоставя на отделните думи подходящи обекти от базата знания и трябва да създаде коректни структури, които показват по какъв начин значенията на отделните думи се комбинират едно с друго.
7/Свързване/ интегриране с контекста: определяне на референтите на местоименията (принцип на най-близкия ляв съсед). Значението на дадено изречение може да зависи смислово от изреченията, които го предхождат. Обектите, въведени в изречението, трябва или да са експлицитно определени, или да се отнасят към обекти от предишни изречения. Целият контекст трябва да бъде ясен и това е целта на тази стъпка.
8/Прагматичен анализ: трансформиране на синтактичното дърво в обобщена команда на вътрешния език на съответната приложна система.

Формализми за описание на *синтаксиса и семантиката* на ОЕЕ
• Разширени мрежи на преходите (Augmented Transition Networks, A TN);
• Граматики, определени от клаузи (Definite Clause Grammars, DCG);
• Семантични граматики (Semantic Grammars) и др.
Проблем: контекстно-свободните граматики нямат достатъчна изразителна сила за описание на синтаксиса на ограничените естествени езици.
СИНТЕЗ НА РЕЧ/ГОВОР:
Основни понятия свързани с речта и нейните елементи:
*Основна честота - най-ниската периодична компонента на речта;
*Фонема - най-малката говорна единица, например В от Варна.
*Алофоните са разновидност на фонемите, например затъмненото б в кораб и б в бор се явяват алофони на фонемата б.
*Морфема - най-малката говорна единица, имаща собствено значение- Пример: млекар ниц а (корен-наставка-окончание);
*Гласов звук - генерира се от гласните струни, например л, м и т.н.;
*Гласни широки и тесни;
*Съгласни звучни и беззвучни.
Могат да се разделят основни три подхода в системите за синтез на говор:
-пряк *синтез* – естествената говорна форма се записва в вид на отделни думи или фрази, след което при зивеждането се пресъздава. На принципа на кодиране с предсказване (прогнозиране) – диферен. имп-кодова модулация.;
- синтез на реч на база анализ – говорния сингал се формира като се извличат негови характерни парам по пътя на анализа, а след това се формира, с помощта на тези парам и цифрова филтрация сингалът се възстановява. Най-често се ползва методът на кодиране с линейно предсказване.;
- При синтеза по правила, отделните фрази и думи се разделят на срички и/или фонеми, които служат за основние стр-рни елементи. Към тях се добавя информация за качествените х-ки на съответния звук. Синтезът се извършва въз осн. база от данни, съдържащи

срички и/или фонеми, алфони, дифони, и правила за формиране на отделните думи. Използва се сливане (конкатенация) на елементи на речта във функция на времето. Компоненти:

В началото се извършва преобразуване на текста в звуков текст. Повечето блокове, изпълняващи тази функция използват речник от правила за преобразуването и изключения, записани в библиотека. Създаването на подобна библиотека е трудоемък процес, но при наличието на диалог тя се усъвършенствува с течение на времето. Потребителят усеща недостатъка в преобразуването и ръчно подобрява произношението.

Във втория блок звукът се преобразува в реч.

Докато в първия блок решенията са предимно програмни, то във втория блок решението е предимно апаратно с използване на сигнални процесори (DSP Digital Signal Processor). Важно изискване при генерирането на реч е работата в реално време. Ако това условие е спазено, то системата за генерация на звук ще може да преобразува текст в говор почти без предварителна обработка.

33. Разпознаване. Изображение и образ. Роля на датчиците при разпознаването. Роля на признаците при разпознаването. Класификация. Обобщена схема на разпознаващо устр-во. Синтактично /лингв./ разпознаване

РАЗПОЗНАВАНЕ. Образ се нарича информационно отражение на някакъв обект, на негови свойства и връзките между тях. Под свойства се имат предвид формата, големината, цвета, теглото, структурата и т. н. на съответния обект. Така че разпознав на образи (РО) се явява ъвкупност от методи и средства за възприемане на (обектите от) околния свят. Процес на присвояване на дадения обект от изображението на идентификатор съответстващ на неговите описателни (атрибутни) характ-ки.

Типични задачи от областта на разпознаването на образи са: разпознаване на ръкописен и печатен текст; разпознаване на обекти от разстояние, обекти от снимки, класифициране на снимки – класифиц по тяхни х-ки; определяне ориентацият на тримерни обекти;

Всеки обект се характеризира с определено множество от признаци. Всеки признак от своя страна се характеризира с някаква дефиниционна област (дефиниционен интервал). При решаване на задачи за разпознаване обикновено се разглеждат не всички признаци на съответните обекти, а само специално подбрано подмножество на множеството на признаците. Разпознав на образи има пряко отношение към термина КЛАСИФИКАЦИЯ.

Основни термини:

Образ : описанието, на който и да е обект от даден клас.

Клас : множество от образи имащи едни и същи свойства.

Признак : общо свойство за обектите от даден клас.

Класификация на образи : отнасянето на даден образ към даден клас на базата на приети критерии и знание.

Класификц има два аспекта:

а/ класифициране на даден обект, т. е. идентифициране на класа, към който принадлежи, чрез изследване на признаците или структурата на обекта;

б/ създаване на описания на отделните класове от обекти в разглежданата предметна област. В този случай говорим за обучение на съответната разпознаваща система. Използват техники от машинното самообучен.

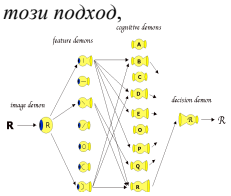
ОБЩАТА структ-функционална СХЕМА на РО (КЛАСИФИКАЦИЯ) на образи се състои от:

сензори, блок за предварителна обработк. Класификация и Памет

сигнали (пристигаща отвън информация във вид на сигнали) -> сензори -> блок за предварителна обработк. -> класификация (с памет отдолу за обучение) -> Клас

Блокът за предварителна обработка включва филтри за изчистване на шумовете, получавани от самите сигнали или поради несъвършенства-та на сензорите. След филтрирането на шумовете се формира множеството от характерните (най-съществените) признаци на разпознавания образ. Блокът памет включва описанията на класовете и други данни, необходими за класификацията. Този блок може директно да се използва или да се самообучава.

Един от основополагащите ПРИНЦИПА за РО е използването на множество от общи признаци. За даден клас wi се определят признаци (x1, xn). Всеки клас се характеризира с n общи признака. За входните образи се определят множество от признаци и те се сравняват с множеството от признаци на всеки един клас. *Пандемониумът на Селфридж използва този подход*,

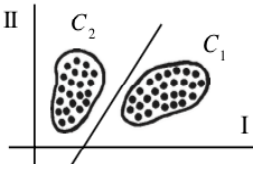


Посъщество е невронна четирислойна (разпределена) мрежа, чиито елементи се наричат демони.

На най-ниското равнище се намират демони на данните (демони на изображението), които играят ролята на сензори. На най-високото равнище се намира демонът на решението, който определя изхода на системата (класифицира разпознатия образ). На равнището под това на демона на решението се намират демони на разбирането. Всеки от тях съответства на един от разпознаваните класове (кривите на буквата R примерно). Между демоните на данните и демоните на разбирането се намират изчислителни демони, които по същество извършват предварителната обработка на сигналите, получени от сензорите (демоните на данни- те), и формират признаците на съответния образ. Всеки демон на разбирането трябва да определи степента, в която образът, получен от демоните на данните, съответства на категорията, която се представя от съответния демон на разбирането. Колкото степента на съответствие е по-висока, толкова по-силен сигнал се изпраща към демона на решението, който избира най-силния от постъпилите към него сигнали.

Основни ПОДХОДИ /Методи за РО

а/ статистически / математически (свежда се до сравн-не на признаците на дадения обект с признаците на вече известни обекти или с признаците, характерни за съответния клас). посочват конкретни формализми за класификация на образите. За разпознаването се използва следния ПРИНЦИП:



Има множество от обекти, които подлежат на класификация. Всеки обект се характеризира с набор от съществени признаци. На базата на използваните признаци се определя признаковото пространство (пространството на признаците). Всеки обект е точка в това пространство. Пример. Дадено е двупризнаково пространство. С всяка от осите на координатната система се свързват стойностите на определен признак.; Такива статист модели методи са: Наивен Бейсов модел. Учене, основано на примери. Алгоритъм на К-най-близък съсед

б/ Структурно лингвистични методи (*синтактични*) (свежда се до изследване на структурата на обектите): за образите се отделят структурни признаци на базата, на които се прави описание на образа. Описанието се анализира с цел извличане на конкретна информация или разпознаване на входния образ. Използва се Пандемониумът на Селфридж при който се ползва обучението на съответната система за РО като процес на постепенно усъвършенстване на алгоритъма за разделяне на обектите (разделящата функция).

За целта системата трябва да разполага с:

*множество от данни (описания на обекти);
*множество от обучаващи примери, което трябва да е пълно - да включва примери за обекти от всички класове;
*множество от тестови примери, които служат за обратна връзка (сечението им с обучаващите примери трябва да е празно).
ОБУЧЕНИЕТО при СРО може да става с индуктивно машинно самообучение:
а/ с учител: учителя дава правилата, алгоритъма за класификация на обектите по класове. Той също посочва и имената на класовете. В случаите на обучение класовете са определени предварително. Освен с обектите{x} (под формата на вектори) системата разполага и с класовете{Ci}.Тук задачата се свежда до уточняване и оптимизация на разделящата функция (например на коефициентите wi). Процесът на уточняване е итеративен и ако се окаже, че не е сходящ, трябва да се разшири множеството на обучаващите примери.
б/ Без учител : на базата на множество образи, които постъпват на входа на системата (обучаващо множество), тя търси характерни особености, свойства. И на тяхна база да раздели входното множество на класове. За класовете назначава номера. При него не е известно колко и какви са класовете - известни са само обектите {x } (вектори). Системата сама се опитва да групира обектите в класове. Тава се нарича кластеризация. В този случай резултатът от работата на системата съществено зависи от избрания метод и от подбора на признаците на обектите.

34. Анализ и синтез на изречения на естествен език. Превод от един естествен език на друг.

35. Същност на конекционалисткия подход. Видове невронни мрежи. Модели и архитектури. Особености на невронните мрежи.

36. Обучаващи правила в невронни мрежи. Обучение и самообучение при невронни мрежи. Алгоритми.

1.Избират се съвсем произволна съвкупност от центрове (независи от това как са избрани тези центрове на резултата).
2.Определят се областите в които влизат точните близки към избраните центрове.
3.Всяка група в която средното разстояние между точките превишават предварително избран праг $б_1$ се дели на две.
4.Определят се нови средни точки за всяка област.
5.Изчислява се разстоянието между всяка двойка средни точки.
6.Обединяват се областите чиито средни точки са на разстояние по-малко от $б_2$. Тази процедура се повтаря докато областите се стабилизират.

37. Персептронни невронни мрежи. Модели и архитектури. Алгоритми. Приложения.

38. Невронни мрежи на Хопфийлд. Модели и архитектури. Алгоритми. Приложения.

39.Рекурентни невронни мрежи. Машини на Болцман. Самообучаващи се мрежи. Приложения.

40. Самоорганизиращи се невронни мрежи. Теория на адаптируемия резонанс. Мрежи на Кохонен. Приложения.

сдсдссдссдсдсдсдсдсдсдсд сдсдсдсдсдсдсдсссдсдсдсд сд