3.6.1. Приведите определение экспертной системы.

Под экспертной системой (ЭС) понимают программную систему, аккумулирующую знания эксперта в определенной области и вырабатывающую решения и рекомендации на уровне эксперта.

3.6.2. Перечислите типовые задачи, решаемые с помощью ЭС.

Перечень типовых задач, решаемых ЭС в самых различных областях, включает:

- интерпретацию — извлечение информации из первичных данных (распознавание образов, определение состава вещества и др.);

- диагностику — обнаружение неисправностей и причин их появления в некоторой системе (медицинской, механической, электронной и др.);

- мониторинг — непрерывная интерпретация данных в реальном времени с сигнализацией о выходе тех или иных параметров за допустимые пределы (контроль движения транспорта, наблюдение за состоянием энергетических объектов и др.);

- прогноз — предсказание вероятных последствий на основе прошедших и настоящих событий (предсказание погоды, прогноз ситуаций на финансовых рынках и др.);

- планирование — определение последовательности действий, направленных на достижение заранее поставленных целей (планирование поведения роботов, составление маршрутов движения транспорта и др.);

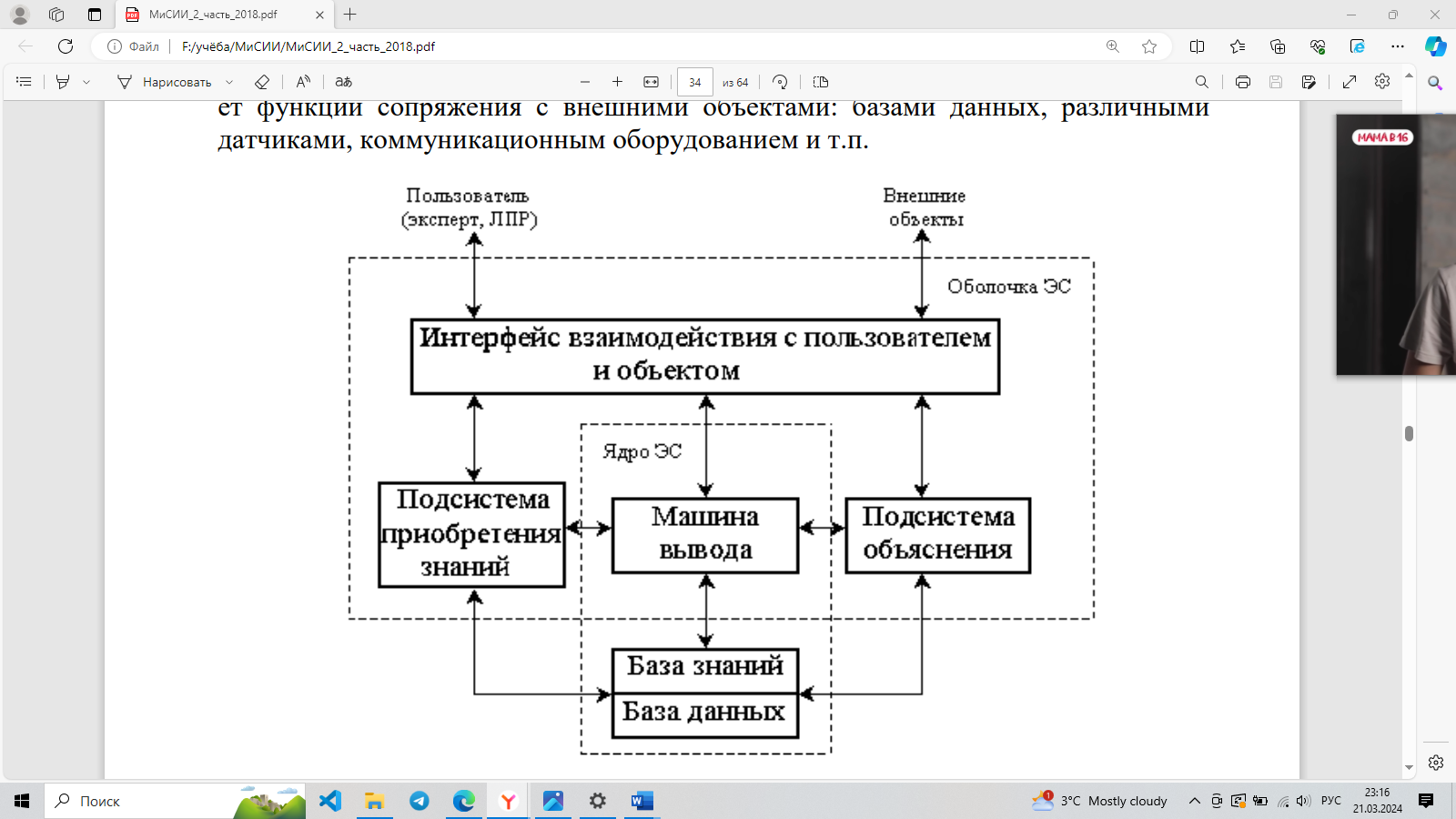
- проектирование — определение конфигурации системы при заданных ограничениях (синтез электронных схем, оптимальное размещение объектов в ограниченном пространстве и др. );

- отладку и ремонт — выполнение последовательности действий по приведению той или иной системы к требуемым режимам функционирования (помощь при отладке программного обеспечения, ремонт инженерных коммуникаций и др.);

- обучение — интерпретация, диагностика и коррекция знаний и умений обучаемого;

- управление — формирование управляющих воздействий, определяющих поведение сложных систем (управление воздушным транспортом, военными действиями, деловой активностью в сфере бизнеса и др.).

3.6.3. Назовите основные компоненты ЭС и объясните их функции.



Ядро ЭС образует база данных, база знаний и машина вывода.

База данных представляет собой рабочую память, в которой хранятся текущие данные, заключения и другая информация, имеющая отношение к анализируемой системой ситуации. База знаний обеспечивает хранение знаний, представленных с помощью одной из моделей: логической, продукционной, фреймовой, сетевой и др.

Машина вывода (подсистема поиска решений), используя данные и знания, организует управление выводом в соответствии с используемой моделью представления знаний. Целью вывода является получение заключений, согласующихся с той информацией, которая содержится в базе знаний.

Подсистема объяснения ЭС позволяет пользователю выяснить, как система получила решение задачи, и какие знания были при этом использованы.

Подсистема приобретения знаний используется как с целью автоматизации процесса наполнения ЭС знаниями, так и при корректировке базы знаний, при ее обновлении, пополнении или исключении элементов знаний.

3.6.4. Назовите основные этапы разработки ЭС, перечислите задачи, решаемые на каждом из этапов.

Идентификация:

Определение проблемной области и задачи.

Поиск эксперта, готового сотрудничать при решении проблемы.

Назначение коллектива разработчиков.

Концептуализация:

Определение предварительного подхода к решению проблемы.

Анализ расходов и прибыли от разработки.

Подготовка подробного плана разработки.

Формализация:

Определение структуры базы знаний.

Разработка формальных правил и фактов.

Создание инференционного механизма.

Наполнение:

Заполнение базы знаний фактами и правилами.

Тестирование и отладка системы.

Опытная эксплуатация:

Внедрение экспертной системы в реальное окружение.

Обучение пользователей.

Мониторинг работы системы.

Тестирование:

Проверка корректности работы системы.

Оценка эффективности и точности решений.

Внесение корректировок и улучшений.

3.6.5. Объясните термины “приобретение знаний” и “извлечение знаний”.

Приобретение знаний – процесс наполнения знаний ЭС

Извлечение знаний – процесс, при котором ЭС использует знания для принятия решений.

3.6.6. Сформулируйте обобщенный алгоритм прямого вывода на правилах продукциях.

Прямой вывод начинается с задания исходных данных решаемой задачи,которые фиксируются в виде фактов в рабочей памяти системы. Правила, предпосылки которых сопоставимы с исходными фактами, обеспечивают генерацию новых фактов, добавляемых в рабочую память. Процесс применения правил к новым фактам продолжается, пока не будет получено целевое состояние рабочей памяти.

Прямой вывод (forward chaining): Прямой вывод начинается с известных фактов и применяет правила-продукции для получения новых фактов или выводов. Он используется, когда у нас есть набор фактов и мы хотим определить, какие выводы могут быть сделаны на основе этих фактов. Алгоритм прямого вывода последовательно активирует правила-продукции на основе текущего состояния фактов и добавляет новые факты, пока не достигнут требуемые условия или пока не будут исчерпаны все возможные правила.

3.6.7. Сформулируйте обобщенный алгоритм обратного вывода на правилах-продукциях.

Обратный вывод, т.е. вывод, управляемый целевыми условиями, начинается с внесения целевого утверждения в рабочую память. Затем отыскивается правило-продукция, заключение которого сопоставимо с целью. Условия применения данного правила помещаются в рабочую память и становятся новой подцелью. Процесс повторяется до тех пор, пока в рабочей памяти не будут найдены необходимые факты, подтверждающие все подцели.

Обратный вывод (backward chaining): Обратный вывод начинается с конечной цели или вывода и работает в обратном направлении, определяя, какие факты или правила должны быть истинными, чтобы достичь этой цели. Он используется, когда у нас есть конечная цель или вывод, и мы хотим определить, какие факты нужны для подтверждения этой цели. Алгоритм обратного вывода рекурсивно проверяет правила-продукции, начиная с цели, и определяет, какие факты или правила необходимы для подтверждения цели. Затем он движется назад по цепочке правил до тех пор, пока не будут найдены все необходимые факты или правила.

3.6.8. Объясните суть вопросов “почему” и “как”, используемых для формирования объяснений вывода в ЭС.

Для получения объяснений выводов пользователи ЭС могут задавать вопросы двух типов:

- почему система сочла необходимым задать пользователю определенный вопрос;

- как система пришла к соответствующему заключению.

3.6.9. Объясните на примере механизм формирования ответа на вопрос типа “как”.

Чтобы ответить на вопрос “как” в системе сохраняется дерево решений,принятых в течение сеанса работы. Просматривая дерево, можно выяснить, достижение каких подцелей привело к данному решению. Для этого после предъявления решения пользователю система может объяснить, как она его получила.

3.6.10. Объясните на примере механизм формирования ответа на вопрос типа “почему”.

Таким образом, формирование ответа на вопрос “как” соответствует просмотру дерева решений в направлении от целевого утверждения до листьев дерева, соответствующих введенным фактам.

Сложнее обстоит дело при прямом выводе, т.е. выводе, управляемом данными. Прямая цепочка рассуждений обеспечивает пользователя менее полезной информацией. Обусловлено это тем, что на промежуточных этапах вывода трудно судить, куда ведет цепочка рассуждений. Так, в ответ на вопрос “почему” пользователю демонстрируется текущее правило. Дальнейшие объяснения не могут быть получены, пока не выполнится следующее правило. Кроме этого, сложно формировать полные ответы на вопрос “как”, даже после достижения цели.

Информация, которая предоставляется, ограничивается списком выполненных правил.

3.6.11. Как можно представить базу продукционных правил на языке Пролог?

В виде предикатов.

3.6.12. Приведите пример простейшей реализации ЭС на языке Пролог.

*Простейший способ* построения ЭС на Прологе — использование в качестве интерпретатора механизма поиска решений, заложенного в Пролог-системе [1,2,4]. Проиллюстрируем это на примере задачи диагностики неисправности электрической плиты. Представим правила, изображенные на рис. 3.3., в виде предикатов языка Пролог:

**/\* структура правила: причина:**− **неисправность \*/**

**нагреватель(неисправен):**− **лампа(светится), плита(холодная). %правило 1**

**выключатель(не\_включен):**− **тока(нет). %правило 2**

**напряжения(нет):**− **тока(нет). %правило 3**

**тока(нет):**− **плита(холодная), лампа(не\_светится). %правило 4**

**лампа(неисправна):**− **лампа(не\_светится), плита(горячая). %правило 5**

Определить причину неисправности плиты можно, указав перечень гипотетических неисправностей и обеспечив проверку выполнимости каждой из гипотез:

|  |  |
| --- | --- |
| **лампа(не\_светится). плита(холодная).** | **% признаки неисправности** |

**гипотеза (нагреватель(неисправен)). % гипотезы**

**гипотеза (выключатель(не\_включен)).**

**гипотеза (напряжения(нет)). гипотеза (лампа(неисправна)).**

**найти(Х):** − **гипотеза (Х), X. % проверка гипотез/**

3.6.13. Какие правила положены в основу предиката найти(Н), осуществляющего обратный вывод на множестве продукционных правил?

Реализуем интерпретатор ЭС в виде предиката найти(Н), где Н - возможная гипотеза, которую требуется подтвердить или опровергнуть. При этом возможны четыре случая:

1) гипотеза Н подтверждается фактом, уже известным системе;

2) гипотеза Н соответствует следствию одного из правил, тогда для подтверждения гипотезы необходимо доказать справедливость предпосылок правила;

3) гипотеза Н, доказываемая на некотором шаге вывода, представляет собой конъюнкцию условий правила, т.е. Н1 и Н2, тогда необходимо доказать достижимость конъюнкции подцелей: найти(Н1) и найти(Н2);

4) гипотеза Н сопоставима с одним из признаков, на основе которых устанавливаются причины неисправности, тогда необходимо задать соответствующий вопрос пользователю.

3.6.14. Напишите на языке Пролог простейший вариант реализации предиката найти(Н).

**найти(H):- Факт :: H. % случай 1**

**найти(H):- Правило :: если H1 то H, найти(H1). % случай2**

**найти(H1 и H2):-найти(H1), найти(H2). % случай 3**

**найти(H):- запрашиваемая(H), % случай 4**

**not(сообщено(H,\_)), % вопроса не было**

**спроси(H).**

3.6.15. Напишете на языке Пролог фрагмент кода, обеспечивающий вывод

вопросов, задаваемых ЭС, и анализ ответов.

|  |  |
| --- | --- |
| **сообщено(H,да).**  **найти(H):- запрашиваемая(H), *not*(сообщено(H,\_)), спроси(H).** | **% вопрос уже был% случай 4% вопроса не было** |

**запрашиваемая(H):- Факт :: признак(H).**

|  |  |
| --- | --- |
| **спроси(H):- *nl*, *write*(H), *write*('?'), *nl*, *read*(O), ответ(H,O). %обработка ответов**  **ответ(H,да):- *assert*(сообщено(H,да)), !. ответ(H,нет):- *assert*(сообщено(H,нет)), !, *fail*.**  **ответ(H,\_):- *write*('правильный ответ: да, нет'), nl, спроси(H).** | **% вывод вопроса% ввод ответа** |

3.6.16. Напишите на языке Пролог фрагмент кода, обрабатывающий вопросы типа «почему».

**ответ**(H,почему,[]):-!,***write***(' Вы задаете слишком много вопросов'),***nl***, спроси(H,[]).

%случай2: в стеке только первая введенная цель, т.е. доказываемая гипотеза

**ответ**(H,почему,[H1]):-!,***write***('моя гипотеза: '), ***write***(H1),***nl***,спроси(H,[]).

% случай3: если в стеке несколько элементов, то вывод заключения (т.е. подцели)

% и номера текущего применяемого правила

**ответ**(H,почему,[Правило | Стек]):-!, Правило :: если H1 то H2, ***write***('пытаюсь доказать '), ***write***(H2),***nl***,

***write***('с помощью правила: '), ***write***(Правило),***nl***, спроси(H,Стек).