6. Типология знаний и языки представления знаний в графодинамических ассоциативных машинах

В разделе 4 был рассмотрен и описан графовый язык SC, являющийся основой представления знаний в графодинамических машинах. В разделе 5 был описан и рассмотрен язык SCL, являющийся sc-подъязыком, и предназначенный для представления знаний в виде формальных теорий, состоящих из логических высказываний, описывающих свойства стационарных реляционных структур. Раздел 6 посвящён рассмотрению способов представления на языках SC и SCL различных видов знаний, имеющих предметную и прикладную специфику. В этом разделе будет рассмотрено представление на языке SC различных шкал измерения и измеряемых величин, представление информации и описание закономерностей динамических систем, способы описания информационных целей в графодинамических ассоциативных машинах, принципы представления нейросетевых моделей и гипертекстовых информационных конструкций, которые в силу свойств языка SC при таком представлении приобретают уникальные свойства.

Данный раздел может быть использован в качестве учебного пособия по дисциплинам «Модели представления знаний, базы данных и СУБД» и «Нейросетевые модели и нейрокомпьютеры» для студентов специальности «Искусственный интеллект».

6.1. Представление знаний, связанных с понятием измерения

Ключевые понятия и идентификаторы ключевых узлов: число, ***номер***, ***шкала измерения***, ***измерение***, ***pwSet***, ***pwArc***, ***pwEl***, ***pwFuzExpr***.

Рассмотрим одну из возможных (но не единственно возможную **(!)**) теоретико-множественную трактовку семантики чисел. Следует отметить, что изначально число появилось как результат измерения какого-либо параметра (свойства, характеристики) у некоторого объекта. Таким образом, процедура измерения – это установление некоторого соответствия между множеством исследуемых объектов (исследуемых на предмет анализа определенного параметра, свойства, характеристики) и некоторым множеством чисел. Каждый измеряемый параметр (свойство) с формальной точки зрения трактуется как множество всех тех и только тех объектов, которые этим свойством обладают. Конкретное число – это множество знаков всевозможных классов эквивалентности элементов с одинаковым значением измеряемого параметра. Каждому измеряемому параметру поставлено в соответствие множество классов эквивалентности, являющееся фактор-множеством для множества элементов, обладающих этим измеряемым параметром. Если каждое число есть множество и если известны элементы этого множества, то можно ввести знаки числа и связать их парами принадлежности с элементами обозначаемых ими множеств. Процедуры измерения одного и того же параметра могут быть различными (разными могут быть единицы измерения, разными могут быть точки привязки к числовой шкале).

Соответственно этому во множестве знаков всех пар принадлежности, выходящих из знаков чисел, выделяются подмножества, каждому из которых соответствует некоторая конкретная процедура измерения. Такие подмножества дуг принадлежности, выходящих из знаков чисел, будем называть конкретными шкалами измерения. Соответственно этому введём ключевой scb-узел с идентификатором ***"шкала измерения"***, множество знаков всевозможных шкал измерения. Заметим при этом, что конкретная шкала (процедура) измерения может использоваться при измерении нескольких параметров.

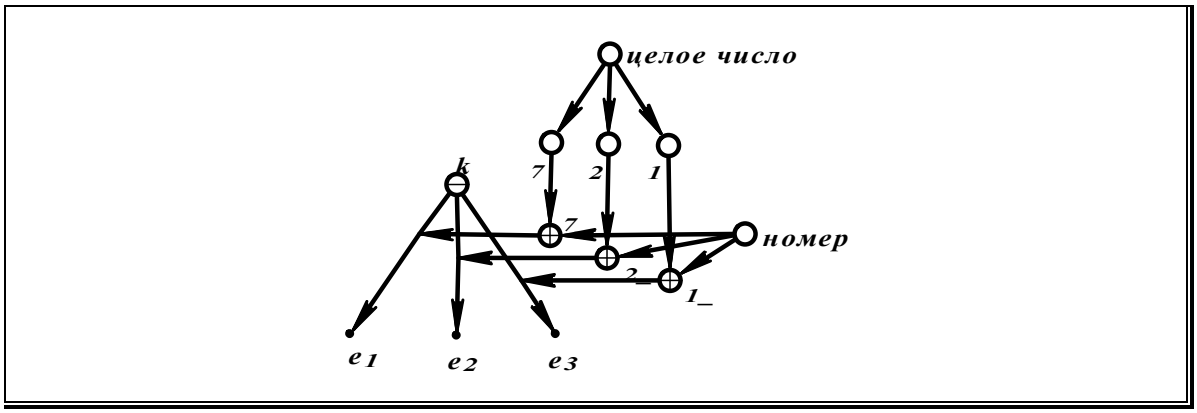
К числу измеряемых параметров, в частности, относятся:

- номер (порядковый номер элемента кортежа);
- мощность множества (количество пар принадлежности, выходящих из знака множества);
- количество элементов (количество элементов множества);
- вес пары принадлежности;
- арность отношения;
- масса;
- температура;
- величина плоского узла;
- расстояние, длина, площадь, объём;
- влажность;
- давление;

- сила электрического тока, напряжение, сопротивление, индуктивность, ёмкость;
- местоположение (координаты на местности);
- отметка времени;
- длительность во времени;
- скорость, ускорение.

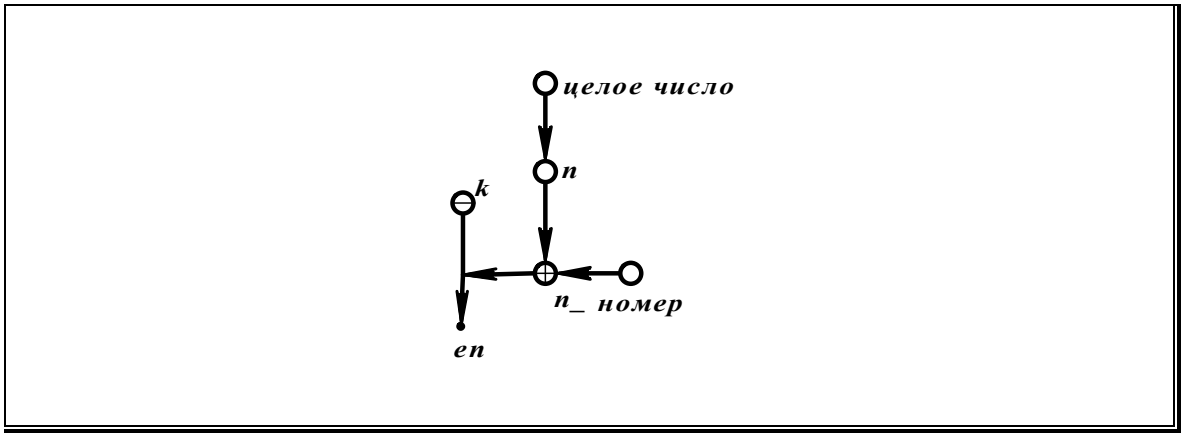
Рассмотрим параметр ***"номер"*** (порядковый номер элемента кортежа). Измеряемым объектом здесь является пара принадлежности, проведенная из знака кортежа. Замена числовых атрибутов (*1_, 2_, ...*) на числа **(!)**, указывающие порядковые номера, дает возможность манипулировать этими номерами, как любыми другими числами. Напомним, что числовые атрибуты, строго говоря, числами не являются. Итак, числовой атрибут ***n_*** есть множество знаков всех тех и только тех пар принадлежности, которые выходят из знаков кортежей и входят в элементы кортежей, имеющие в рамках этих кортежей порядковый номер ***n***.

Приведем пример записи результата "измерения" порядкового номера элемента в кортеже.



Эта конструкция означает, что в рамках кортежа ***k*** элемент ***e1*** имеет **1**-й номер, элемент ***e2*** – **2**-й номер, а элемент ***e3*** – **3**-й номер. Заметим, что числа, указывающие номера элементов в кортежах, могут быть не только натуральными (т.е. положительными целыми числами), но и отрицательными целыми числами. Кроме того, номер элемента кортежа может быть нулевым. Следовательно, числа, указывающие номера элементов в кортежах, в общем случае относятся к классу целых чисел, а не к классу натуральных чисел. В качестве примера см. представление чисел в позиционных системах счисления.

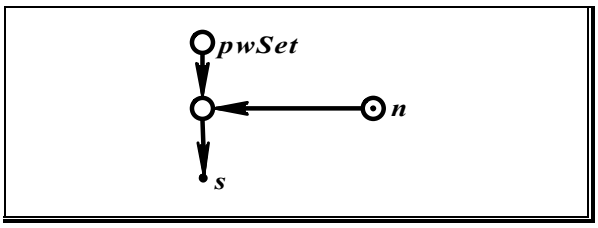
Нетрудно заметить, что между понятием порядкового номера элемента в кортеже и понятием числового атрибута имеет место следующее соотношение.



Такая "замена" числовых атрибутов на числа дает возможность описывать соотношения между номерами элементов кортежей с использованием всего многообразия числовых отношений.

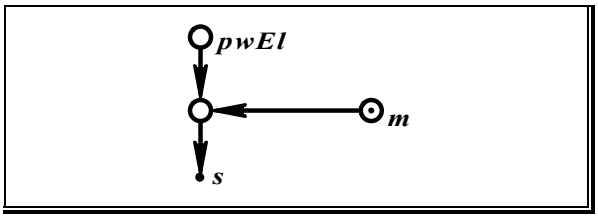
Примечание. Далеко не в каждом кортеже используется нумерация его элементов. То есть элементы далеко не каждого кортежа обладают свойством иметь порядковый номер.

Параметр **“*мощность множества*”** будем также идентифицировать синонимичными идентификатором **“*pwSet*”** (power set). Приведем пример.



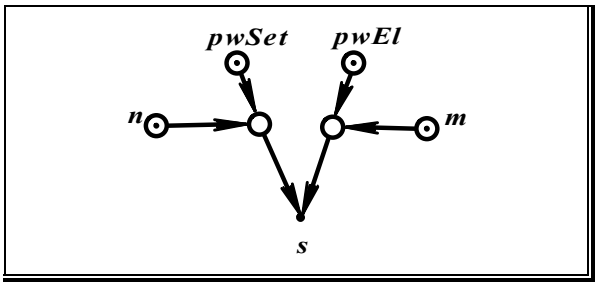
Эта конструкция означает, что мощность (***pwSet***) множества ***s*** равна числу ***n***. При этом число ***n***, очевидно, принадлежит множеству натуральных чисел.

Параметр **“*количество элементов*”** (количество элементов множества) будем также идентифицировать синонимичным идентификатором **“*pwEl* ”** (power elements). Приведем пример.



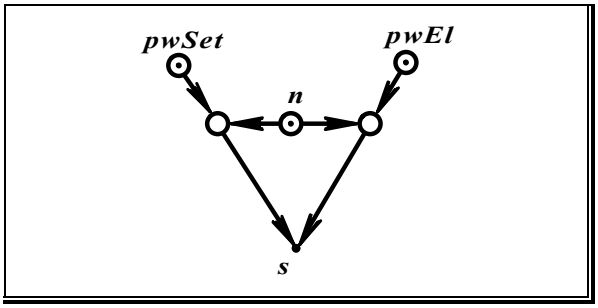
Эта конструкция означает, что количество элементов множества ***s*** равно числу ***m***.

Если множество ***s*** имеет многократное вхождение каких-либо элементов, то имеет место следующая конструкция.



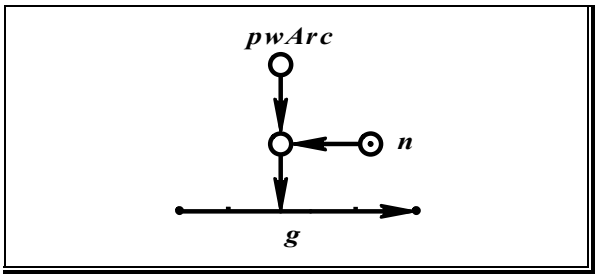
Здесь числа ***n*** и ***m*** не совпадают.

Если же множество ***s*** не имеет кратных элементов (т.е. является канторовским), то имеет место следующая конструкция.



Здесь числа ***n*** и ***m*** совпадают.

Рассмотрим параметр **“*вес пары принадлежности*”** (сила пары принадлежности, мощности scb-дуги, вес scb-дуги), который будем также идентифицировать синонимичным идентификатором **“*pwArc*”**. Приведем пример.

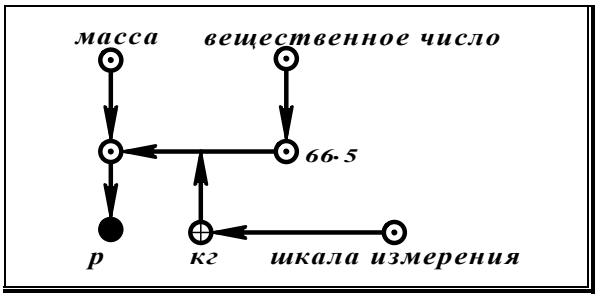


Эта конструкция означает, что вес пары принадлежности ***g*** равен числу ***n***.

При этом:

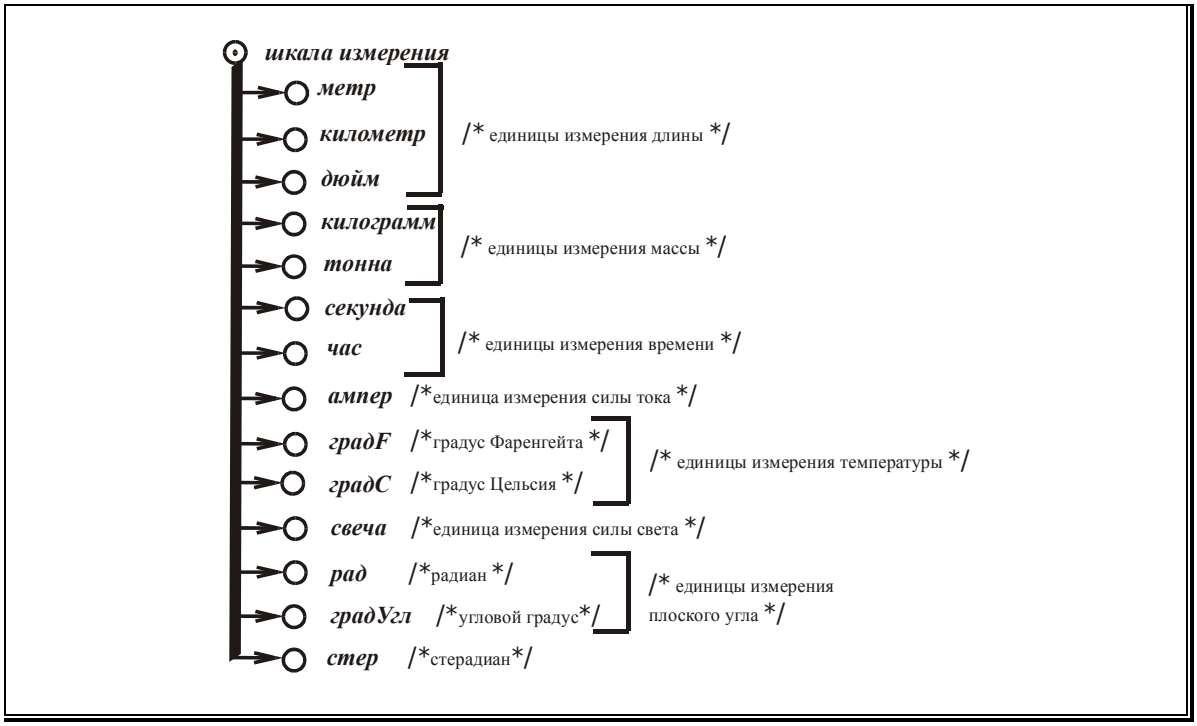
- если $n = 0$, то scb-дуга g негативна;
- если $n = 1$, то scb-дуга g позитивна;
- если $0 < n < 1$, то scb-дуга g считается нечетной с весом n (здесь число n также будем называть степенью нечеткости, степенью достоверности, степенью размерности дуги g);
- если n – целое число, большее 1, то scb-дуга g считается позитивной n-кратной дугой (здесь число n будем также называть кратностью дуги g).

Приведем пример записи результата измерения массы физического тела.



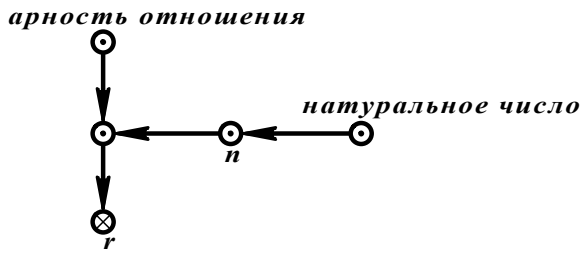
Эта конструкция означает, что масса физического тела p равна 66.5 кг.

Здесь введено ключевое понятие **"*шкала измерения*"** (быть шкалой измерения), которая обозначает множество знаков всевозможных шкал измерения. Заметим при этом следующее. То, что называют единицами измерения, есть простейший вид шкал измерения.



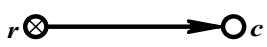
Примечание. Далеко не для всех измеряемых параметров (измеряемых характеристик) необходимо дополнительно указывать шкалу измерения. Это необходимо только тогда, когда измеряемому параметру соответствует несколько **(!)** шкал измерения. Примерами параметров, каждому из которых соответствует единственная **(!)** шкала измерения, являются параметры: $pwSet$, $pwEl$, $pwArc$.

Приведём пример записи результата "измерения" арности отношения.

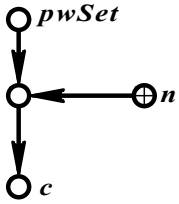


Эта конструкция означает, что отношение ***r*** является ***n***-арным отношением, т.е. представляет собой семейство ***n***-арных множеств (множеств, мощность которых равна ***n***). Заметим, что далеко не каждое отношение обладает свойством иметь арность. Этим свойством обладают те и только те отношения, каждое из которых представляет собой семейство множеств одинаковой мощности. То есть понятие **"*арность отношения*"** и понятие **"*семейство множеств одинаковой мощности*"** являются синонимами.

Нетрудно заметить также, что приведенная выше scb-конструкция эквивалентна утверждению о том, что для каждого элемента с множества ***r***, т.е. для каждой конструкции вида

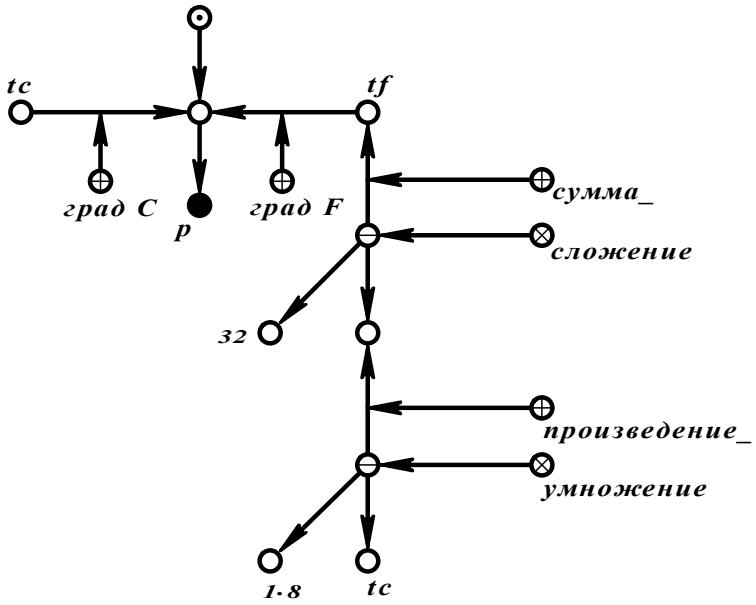


имеет место конструкция вида



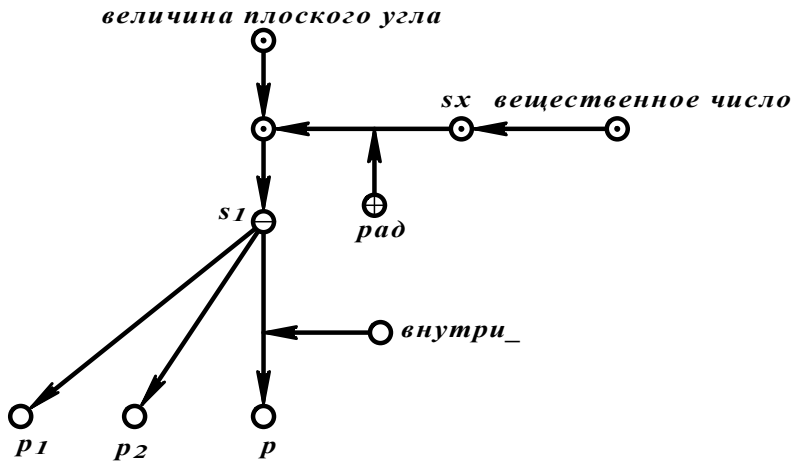
Температурная шкала Цельсия и температурная шкала Фаренгейта связаны между собой следующим соотношением: $tf = 1.8 * tc + 32$,
где
tc – отметка температуры по шкале Цельсия
tf – отметка температуры того же **(!)** предмета ***p***, но по шкале Фаренгейта.

На языке SCB указанное соотношение выглядит следующим образом:



/* *t* – множество всевозможных предметов, имеющих одинаковую с предметом *p* температуру */

Приведём пример записи результата измерения величины плоского угла в радианах. Измеряемым объектом здесь можно считать тернарный кортеж, состоящий из трёх геометрических фигур, лежащих на одной плоскости. При этом две из этих фигур являются либо отрезками, либо прямыми, либо лучами, а третья фигура трактуется как фигура, лежащая внутри измеряемого угла. Собственно измеряемым углом здесь является один из четырех углов, образованных пересекающимися прямыми, на которых лежат указанные выше отрезки или лучи.



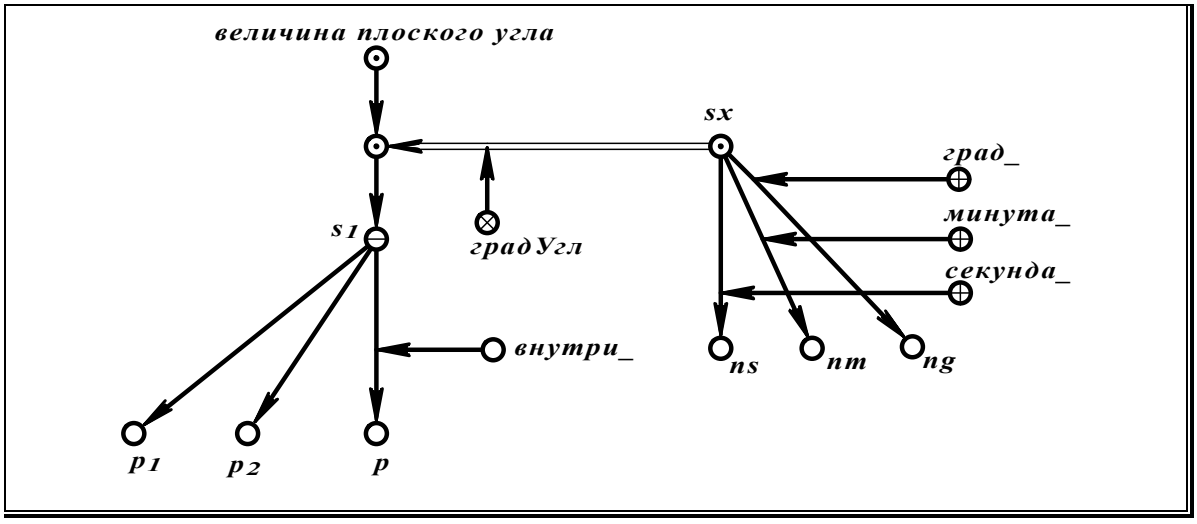
Эта конструкция означает, что число *x* есть результат измерения (в радианах) величины плоского угла, который составлен геометрическими фигурами p_1 и p_2 (каковыми могут быть прямые, лучи, отрезки, множества точек, лежащих на одной прямой точек) и внутри которого находится геометрическая фигура *p*.

Если измерение величины плоского угла осуществляется в угловых градусах, то результатом измерения будет уже не число, а тернарный кортеж чисел, компонентами которого являются:

- целое число в диапазоне от 0 до 360, указывающее количество угловых градусов;
- целое число в диапазоне от 0 до 60, указывающее количество угловых минут;

- целое число в диапазоне от 0 до 60, указывающее количество угловых секунд.

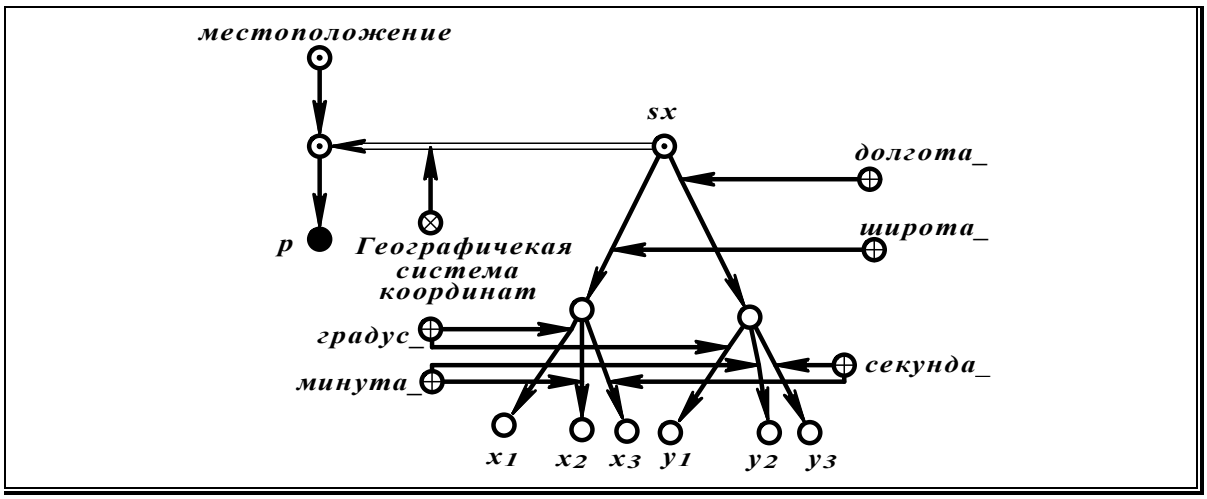
Приведем пример записи результата такого измерения.



Примечание. Особое внимание обратим на то, что результат измерения с парой ***"измеряемый параметр – измеряемый объект"*** связан здесь ориентированной парой, которая парой принадлежности не является. Такого рода результаты измерения будем называть векторными, противопоставляя их рассмотренным выше скалярным.

Приведём несколько примеров записи результатов измерения, когда этими результатами являются векторные величины.

Рассмотрим пример записи результата измерения местоположения некоторого объекта p на земной поверхности в географической системе координат.



Эта конструкция означает то, что местоположение объекта p в географической системе координат определяется широтой, которая задаётся кортежем
□ $градус\_ : x_1$, $минута\_ : x_2$, $секунда\_ : x_3$ □;
и долготой, которая задаётся кортежем
□ $градус\_ : y_1$, $минута\_ : y_2$, $секунда\_ : y_3$ □;
Здесь
x_1 – целое число в диапазоне от -90 до +90;
y_1 – целое число в диапазоне от -180 до +180;
x_2, x_3, y_2, y_3 – целые числа в диапазоне от 0 до 60.

Кроме географической системы координат для измерения местоположения объектов существует большое количество других систем координат.

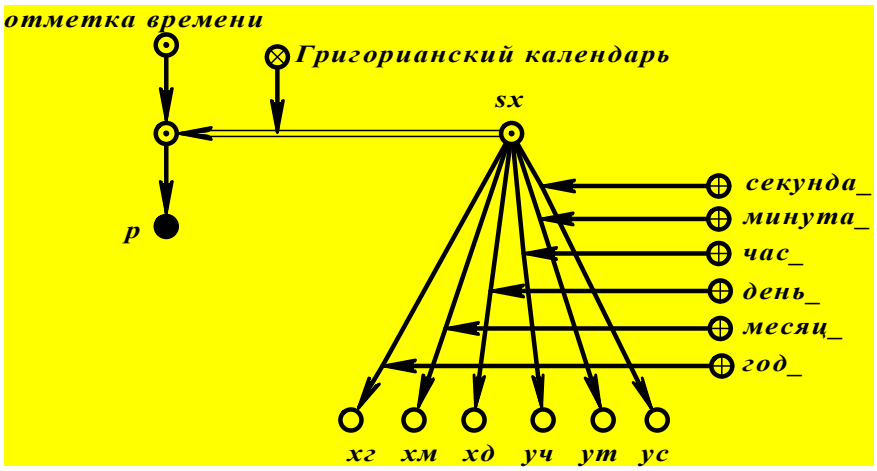
система координат !; ***Географическая система координат***,
Геодезическая система координат,
Горизонтальная система небесных координат,
Первая экваториальная система небесных координат,
Эклиптическая система небесных координат,
Галактическая система небесных координат;

система координат ⊃ ***декартова система координат***;

система координат ⊃ ***сферическая система координат***;

система координат ⊃ ***цилиндрическая система координат***;

Рассмотрим пример записи результата измерения отметки времени для некоторого события (процесса, ситуации).



Пусть $xг = 1999$, $xм = 12$, $xg = 15$, $xч = 14$, $xт = 30$, $xs = 30$.

Тогда приведённая конструкция означает то, что событие ***p*** произошло 15 декабря *1999* года в *14* часов *30* минут *30* секунд.

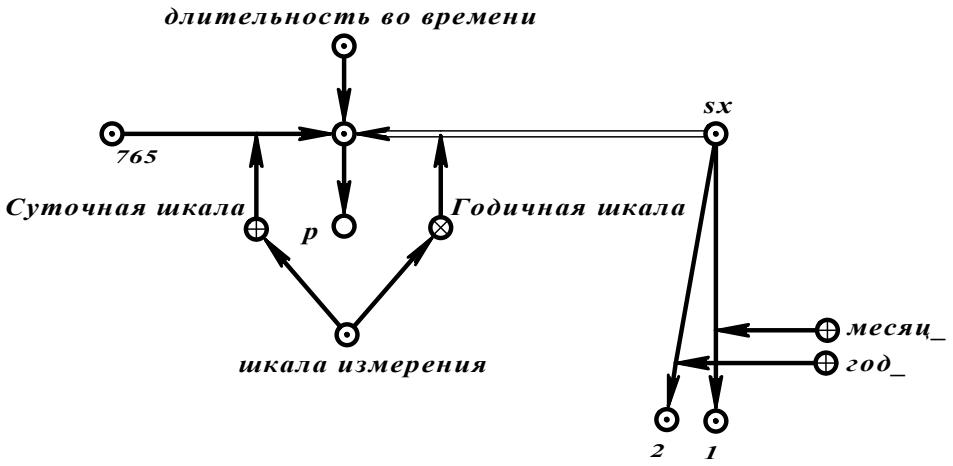
Для рассматриваемой конструкции:
xг представляет собой целое число в диапазоне от –∞ до +∞,
xм – целое число в диапазоне от *1* (январь) до *12* (декабрь);
xg – целое число в диапазоне от *1* до *31*;
xч – целое число в диапазоне от *0* до *23*;
xт, ***xs*** – целые числа в диапазоне от *0* до *59*.

Кроме григорианского календаря известны и другие шкалы измерения отметки времени:

шкала измерения отметки времени !; ***Григорианский календарь***,
Юлианский календарь;

Параметр "отметка времени" на некоторой шкале времени не следует путать с параметром "длительность во времени", который характеризует отрезок времени, в течение которого происходит некоторое событие, или, образно говоря, характеризует период "существования" (время "жизни") указанного события.

Приведём примеры записи результата измерения длительности во времени (отрезка времени).



Приведённая запись означает, что некий процесс ***p*** длился *2* года *1* месяц или (по другой шкале) *765* суток (напомним, что года могут быть високосными, а месяцы могут иметь разное количество дней).

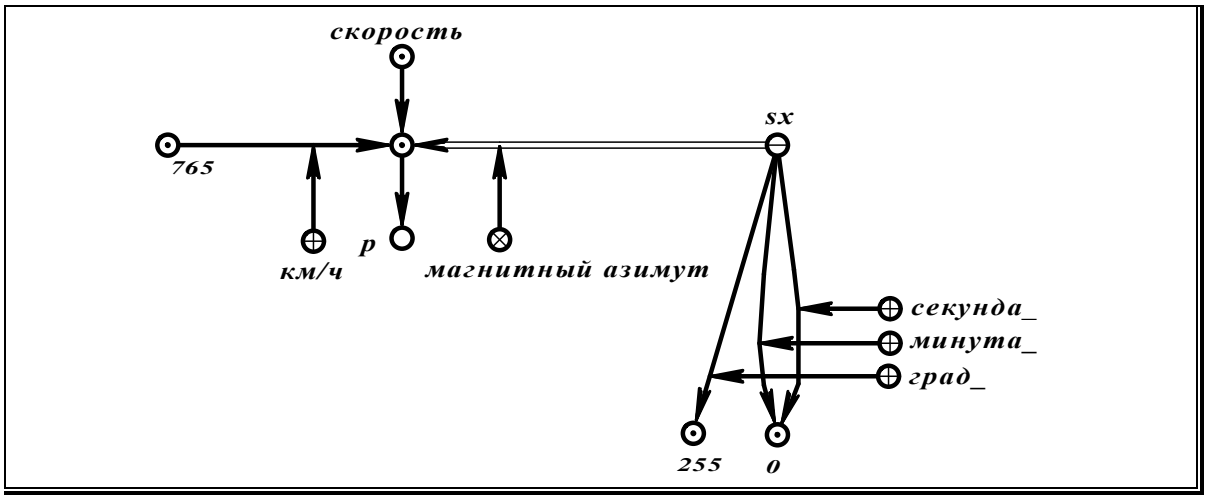
Всё, о чём говорилось выше, имеет отношение к изменению скалярных **(!)** параметров. Но кроме скалярных параметров, есть векторные параметры, результатом измерения которых являются не числа, а кортежи **(!)** чисел.

Заметим то, что результатом измерения одного и того же параметра в зависимости от шкалы измерения может быть как скалярная величина, так и векторная величина (кортеж чисел).

В приведённой конструкции введено понятие шкалы измерения времени (шкалы измерения длительности во времени).

```
шкала измерения времени !;      Годичная шкала ,
                                 Суточная шкала ,
                                 час /* измерение длительности в часах */ ,
                                 минута ,
                                 секунда , миллисекунда , наносекунда ;
```

Приведём пример записи результата измерения параметра, который характеризуется не только величиной, но и направлением. Примером такого параметра является **скорость**. Рассмотрим запись утверждения о том, что некий объект p в некий (неуточняемый) момент времени движется со скоростью *72* километра в час (км/ч) в направлении на юго-запад. Направление движения на поверхности земли задаётся азимутом (например, магнитным азимутом) и измеряется величиной угла между направлением на точку севера и соответственно направлением движения. При этом отсчёт величины угла осуществляется от направления на север по часовой стрелке.



Здесь при измерении азимута градус задаётся натуральным числом в диапазоне от *0* до *360* (точнее, до *359*), а минута и секунда – натуральным числом в диапазоне от *0* до *60* (точнее, до *59*).

Введём понятие шкалы измерения величины скорости и понятие шкалы измерения направления (в частности направления движения).

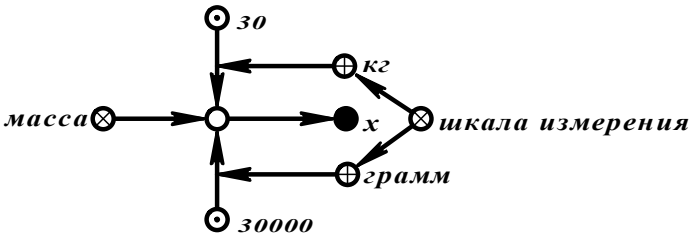
```
шкала измерения величины скорости !;   км/ч ,
                                        м/с ,
                                        узел /* морской */ ;

шкала измерения направления !;   магнитный азимут,
                                 истинный азимут /* географический */ ;
```

Подведя итог вышесказанному, можно ввести обобщённое отношение **"*измерение*"**, которое представляет собой бинарное ориентированное отношение, каждая пара которого связывает:

1) знак пары принадлежности, связывающей знак измеряемого параметра (измеряемой характеристики) со знаком измеряемого объекта (каковым может быть всё что угодно);
2) результат измерения, каковым может быть как число (скалярная величина), так и числовой кортеж (векторная величина).

При этом, если результат измерения является скалярной величиной, то соответствующая пара отношения **"*измерение*"** является парой принадлежности **(!)**.

Отношение **"*измерение*"** разбивается на целый ряд подмножеств, каждому из которых соответствует та или иная **шкала измерения** (в частности, единица измерения). Приведём ещё несколько примеров.

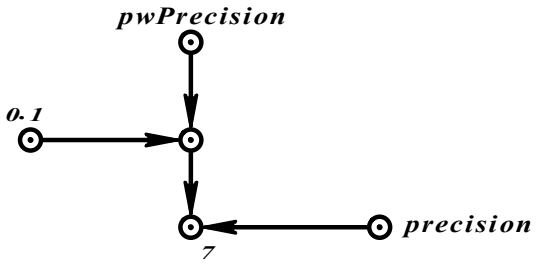


/* некоторый объект имеет массу, равную *30* кг */

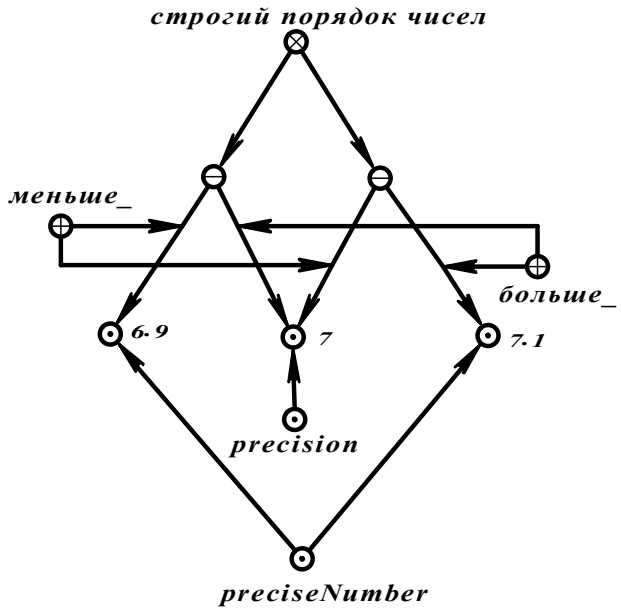
Ключевой узел ***fuzSet*** является знаком унарного отношения "Быть нечетким, неопределенным множеством", т.е. множеством, строгое определение которого отсутствует или, другими словами,

отсутствует четкий критерий, позволяющий установить принадлежность или непринадлежность произвольного sc-элемента этому множеству. Характерным признаком нечетких множеств является наличие большого количества нечетких константных sc-дуг, выходящих из sc-узла, обозначающего нечеткое множество. Хотя, конечно, некоторое количество нечетких константных sc-дуг может выходить из sc-узлов, обозначающих также и четкие множества. Но нечеткость таких дуг обусловлена не отсутствием критериев принадлежности таким множествам, а просто недостаточностью (для соответствующих критериев) сведений о конкретных потенциальных элементах этих множеств. Проведение в некоторый константный sc-узел негативной константной sc-дуги из ключевого узла ***fuzSet*** означает то, что указанный sc-узел обозначает четкое множество. Ключевой узел ***fuzSet*** относится к группе ключевых узлов, используемых для описания теоретико-множественных соотношений.

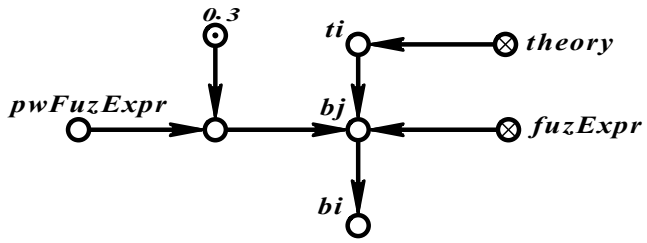
Приведём для сравнения два способа представления точности чисел (с использованием и без использования понятия измерения).



/* Здесь ***precision*** – знак множества всех неточных чисел, степень точности которых может быть дополнительно уточнена, ***pwPrecision*** – фактор-множество, множества неточных чисел по степени их точности */



Оценка степени истинности/ложности нечёткого высказывания осуществляется с помощью фактор-множества *pwFuzExpr* по шкале от *0* до *1*.



/ bj* – высказывание, которое в рамках теории *ti* считается нечётким со степенью истинности *0.3* **/*

6.2. Описание динамических систем

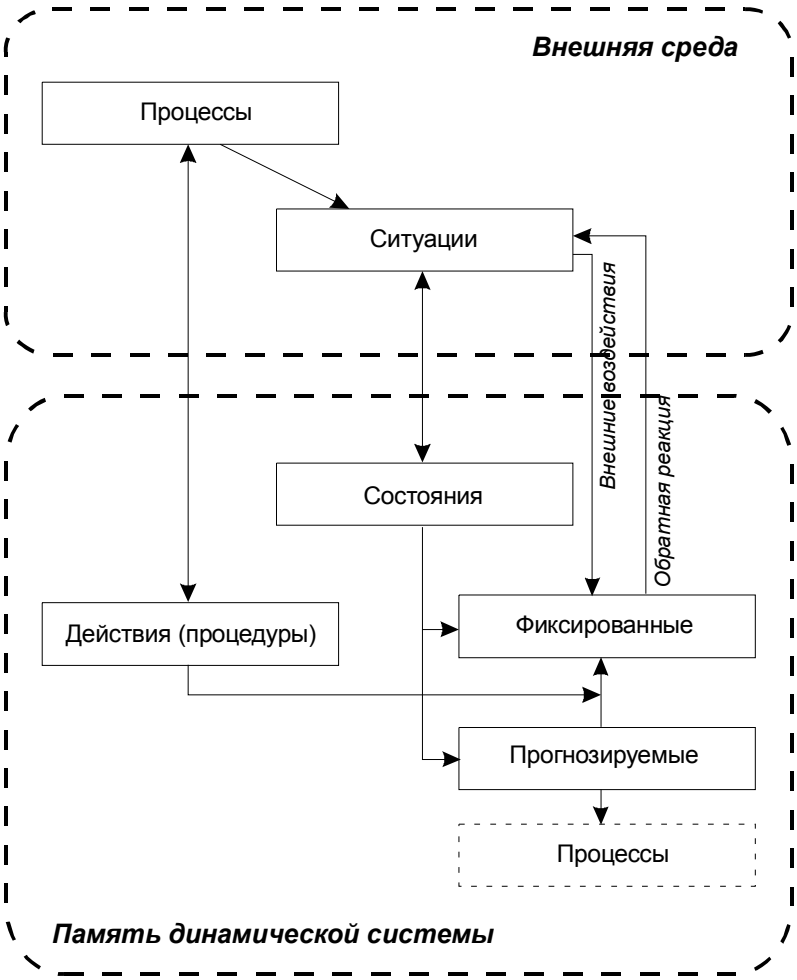
Ключевые понятия и идентификаторы ключевых узлов: процесс, **ситуация**, **действие**, **состояние**, **нестационарная информационная конструкция**, **нестационарная предметная область**, **стационарная константа**, **нестационарная константа**.

Как уже было отмечено, scl-теория является способом описания того или иного состояния немодифицируемой формальной модели переработки знаний. Немодифицируемые формальные модели переработки знаний всегда соответствуют стационарным предметным областям, а различные состояния таких моделей отражают различные состояния процесса решения задач в рамках этих моделей, т.е. задач, формулируемых по отношению к указанным стационарным предметным областям. В отличие от этого, в ходе выполнения семиотической (т.е. модифицируемой) модели переработки знаний осуществляется не только решение задач в рамках той или иной немодифицируемой формальной модели, но и порождение новых немодифицируемых формальных моделей путем того или иного преобразования имеющихся формальных моделей. Нетрудно заметить, что описать состояние семиотической модели на языке SCL - это значит построить некоторую scl-метатеорию, описывающую некоторое множество scl-теорий, систему связей между этими scl-теориями, а также всевозможные правила их преобразования.

Одним из примеров такой scl-метатеории является описание нестационарной предметной области, в основе которого лежит трактовка нестационарной предметной области как иерархической системы квазистационарных предметных областей, называемых состояниями (или ситуациями) нестационарной предметной области. Нестационарные предметные области имеют как стационарные свойства, для представления которых, в частности, используются стационарные (неситуативные) связки и отношения, так и нестационарные свойства, зависящие от состояния.

Прежде чем перейти к рассмотрению средств языка SCL для создания динамических систем, приведем некоторые основные используемые понятия. Функционирование динамической системы будем рассматривать как процесс взаимодействия между внешней средой и рабочей памятью динамической системы (см. рис.6.2.1). В рамках внешней среды будем различать **процессы** и **ситуации**. Таким образом, процессы, которые в памяти динамической системы будем трактовать как некоторые **действия** или процедуры, порождают соответствующие ситуации, которые в памяти динамической системы будем представлять в виде **состояний** этой системы. Таким образом, процесс может задаваться в виде последовательности ситуаций и элементарных действий, приводящих к этим ситуациям. Тогда процесс в рамках динамической системы будем трактовать как выполнение заданных действий между состояниями. Кроме того, описания в памяти динамической системы некоторых состояний будем также называть процессами в том случае, если это описание некоторых конкретных действий системы, направленных на преобразование ее состояний.

Р и с у н о к 6 . 2 . 1 . Схема функционирования динамической системы



Рассмотрим основные средства описания динамических систем (нестационарных предметных областей) средствами языка SCL. В качестве рабочей памяти динамической системы будем рассматривать sc-память, в которой хранятся и обрабатываются соответствующие scl-конструкции.

Знание о нестационарной предметной области на языке SCL представляется путем его расчленения на множество знаний о квазистационарных предметных областях, каждое из которых описывает некоторое состояние описываемой предметной области, трактуя нестационарную предметную область как стационарную на некотором промежутке времени по отношению к указываемым свойствам. Описание каждого такого состояния оформляется как scl-теория, являющаяся стационарным компонентом (state-компонентом) scl-метатеории, описывающей нестационарную предметную область в целом. Для задания таких scl-метатеорий вводится специальное отношение, обозначаемое ключевым узлом ***theoryDyn***.

Итак, sc-конструкция вида:

theoryDyn !; ***ki*** !; ***state_*** **:** ***bj*** **;**

семантически означает, что ***bj*** есть высказывание, описывающее некоторое состояние (некоторую ситуацию) той нестационарной предметной области, которая описывается теорией ***ki*** . Высказывание ***bj*** может быть либо атомарным, либо конъюнктивным. Элементы атомарного высказывания ***bj*** и элементы union-компонента конъюнктивного высказывания ***bj*** могут быть как константными, так и переменными. При этом, если такой константный sc-элемент является также константой scl-метатеории (элементом union-компонента высказывания ***ki*** , если ***ki*** <! ***theoryDyn*** **;**), то он называется **стационарной константой**. Если же указанный константный sc-элемент (как узел, так и дуга) не является константой scl-метатеории, то он называется **нестационарной** (ситуативной) **константой**.

Таким образом, state-компонент есть перечисление некоторых свойств, в частности, ситуативных связей нестационарной предметной области, которые сохраняются в течение некоторого отрезка времени. Следовательно, каждый state-компонент, не имеющий свободных переменных, являясь знаком соответствующего высказывания, однозначно соответствует квазистационарному процессу, описываемому этим высказыванием. Каждому такому процессу ставится в соответствие отрезок времени его существования с моментами начала и конца процесса.

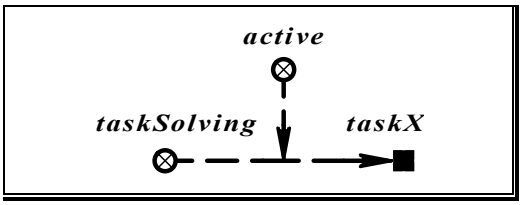
Каждое высказывание, являющееся state-компонентом некоторого другого высказывания, в свою очередь, само может также иметь несколько state-компонентов, которые осуществляют разбиение некоторого состояния (процесса) на несколько более "мелких" процессов. Если некоторое высказывание является state-компонентом и имеет свободные переменные, то оно представляет собой высказывание о существовании соответствующего состояния.

Поскольку моделирование и реализация динамических процессов сводится к анализу в каждый конкретный момент времени некоторой создавшейся к этому времени ситуации, которая определяет текущее состояние системы, необходимо рассмотреть типологию состояний динамической системы. В связи с этим, будем различать следующие типы состояний:

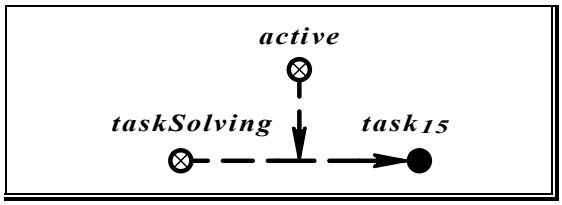
1) **фиксированное состояние** - представляет собой некоторое промежуточное состояние динамической системы, фиксируемое в некоторый момент времени и описываемое некоторой конкретной sc-конструкцией, состоящей из стационарных и ситуативных констант соответствующей scl-метатеории. При описании процессов (правил перехода из одного состояния в другое) совокупность **прогнозируемых** (предполагаемых) **состояний** описывается изоморфными sc-конструкциями, в которых ситуационные константы обозначаются переменными sc-элементами. Так, например, на sc-тексте 6.2.1 а) представлен пример описания прогнозируемого состояния, свидетельствующего о том, что в некоторый момент времени интеллектуальная обучающая система (ИОС), рассматриваемая как динамическая система, может перейти в состояние активизации стратегии решения задачи. На sc-тексте 6.2.1 б) представлено соответствующее фиксированное состояние, семантика которого заключается в том, что ИОС в данный момент находится в состоянии решения конкретной задачи;

SC-текст 6.2.1. Пример описания на языке SCL а) совокупности прогнозируемых состояний и
б) конкретного фиксированного состояния

а)



б)

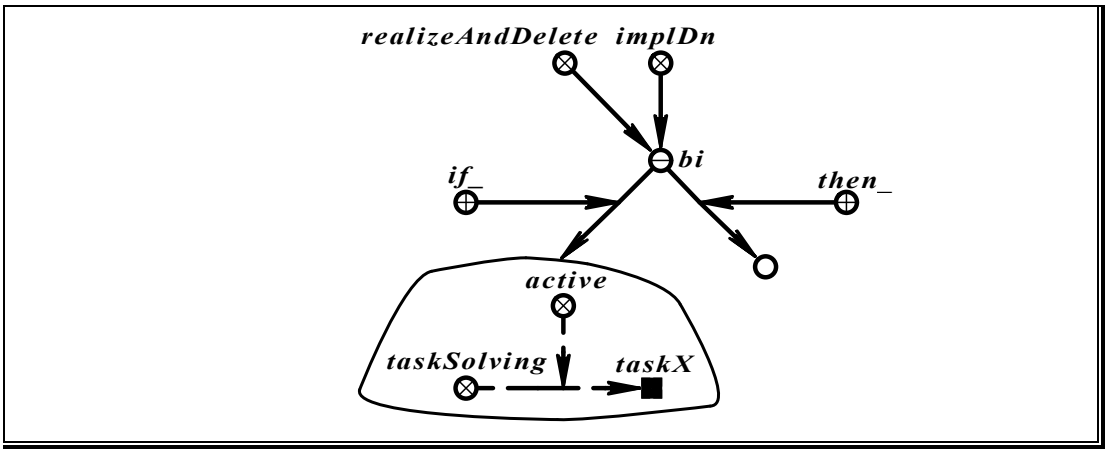


2) **состояние перехода** (переходное состояние). Описание переходного состояния представляет собой правило перехода из одного фиксированного состояния в другое. На языке SCL для описания переходных состояний используются ключевые узлы ***implDn*** , ***transfDn*** , ***transfDnW*** , ***if_*** , ***then_*** , ***worker_*** , описание семантики которых будет дано ниже. При формальном описании на языке SCL будем также различать **общее описание переходного состояния** и **частное описание переходного состояния.** Общее описание представляет собой scl-высказывание, которое постоянно хранится в sc-памяти и может быть применено (зафиксировано) многократно. Частное описание генерируется в sc-памяти в некоторый конкретный момент времени в результате каких-либо действий системы и после однократного применения удаляется. Для выделения множества частных описаний переходных состояний в рамках языка SCL используется соответствующее унарное отношение ***realizeAndDelete.*** Таким образом, если ***bi*** – знак некоторого состояния, то наличие sc-конструкции вида:

realizeAndDelete !; ***bi*** ;

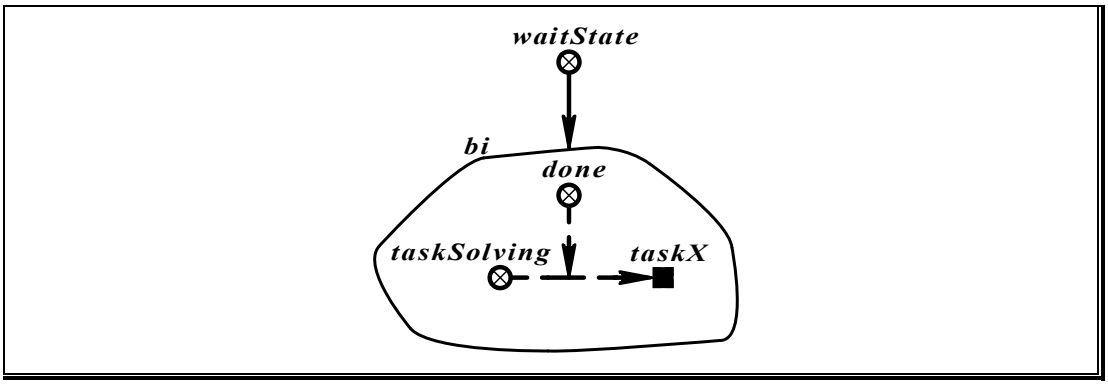
означает, что после фиксации ситуации, соответствующей описанию состояния ***bi***, это описание будет удалено из sc-памяти. Например, как только оба компонента изображенного на scl-тексте 6.2.2 частного описания состояния перехода ***bi*** преобразуются в фиксированные (см. sc-текст 6.2.1 а)), sc-узелы ***bi*** , ***be*** и ***bt*** будут удалены из sc-памяти вместе с инцидентными им дугами. Состояния перехода будем также называть **процессами**;

SCL-текст 6.2.2. Пример описания на языке SCL частного состояния перехода



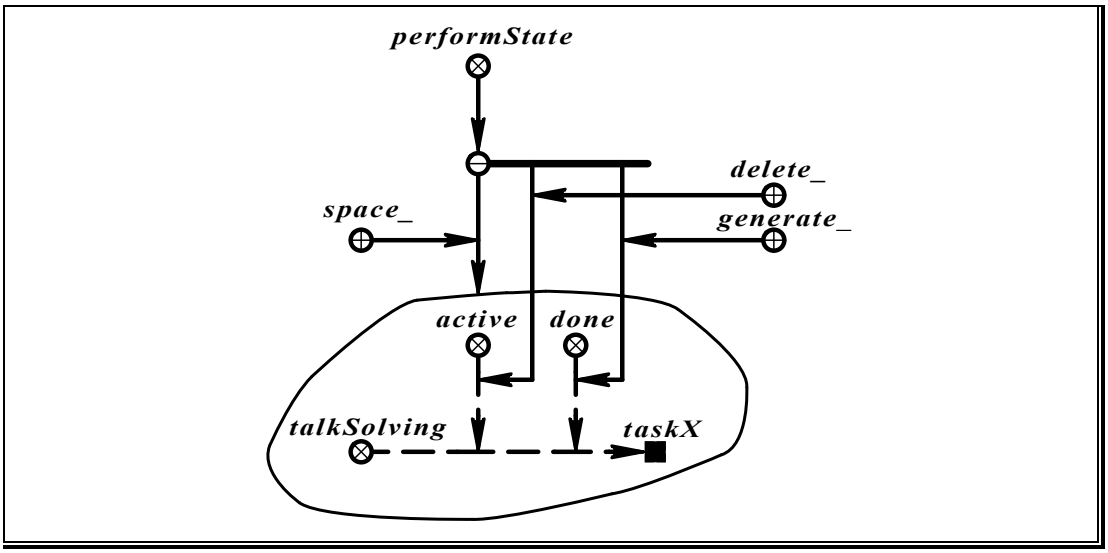
3) **состояние ожидания** - при описании на языке SCL помечается в составе нестационарной scl-метатеории ключевым узлом ***waitState***. SC-узел, обозначающий состояние ожидания, является знаком sc-конструкции, описывающей ожидаемое состояние фрагмента sc-памяти. Примером состояния ожидания в ИОС является ожидание завершения реализации некоторой стратегии обучения (см. scl-текст 6.2.3);

SCL-текст 6.2.3. Пример описания на языке SCL состояния ожидания

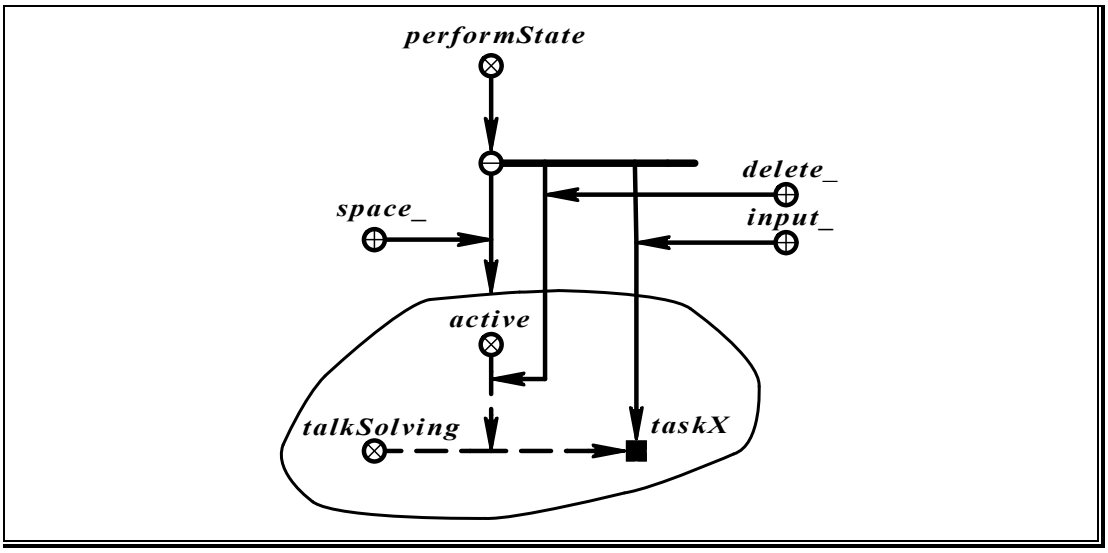


4) **состояние преобразования памяти** - описывается в составе нестационарной scl-метатеории с помощью отношения ***performState*** , которое является отношением нефиксированной арности и имеет атрибуты: ***space_*** - область памяти, которая подлежит преобразованию; ***generate_*** - указывает на элемент, который необходимо сгенерировать в sc-памяти, ***delete_*** - указывает на элемент, подлежащий удалению, ***input_*** - указывает на sc-узел, содержимое которого необходимо загрузить в sc-память. Обработка указанного отношения приведет к реализации процесса по преобразованию заданного фрагмента sc-памяти. На scl-текстах 6.2.4 и 6.2.5 приведены примеры описания состояния преобразования памяти. Смысл scl-текста 6.2.4 заключается в том, что в выделенном фрагменте sc-памяти необходимо удалить sc-дугу, выходящую из узла ***active,*** и сгенерировать sc-дугу, выходящую из узла ***done***. Семантически описание данной операции в составе ИОС означает преобразование стратегии решения задач, обозначенной sc-узлом ***taskSolving***, из активной (***active***) в выполненную (***done***). Обработка scl-текста 6.2.5 будет заключаться в том, чтобы сгенерировать sc-дугу, выходящую из узла ***active*** и входящую в соответствующую дугу (тем самым активизировав стратегию решения задач ИОС), а также загрузить в память содержимое sc-узла ***taskX*** (в котором в виде sc-текста может храниться условие подлежащей решению задачи).

SCL-текст 6.2.4. Пример описания на языке SCL состояния преобразования памяти

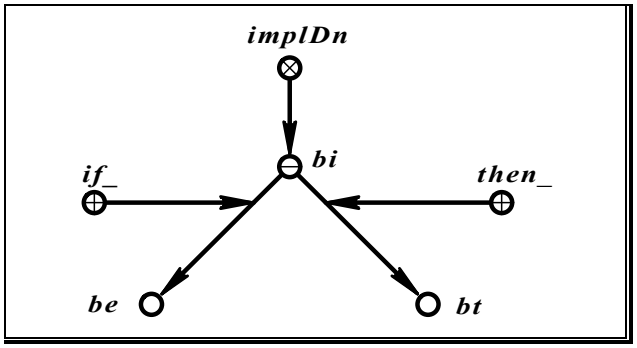


SCL-текст 6.2.5. Пример описания на языке SCL состояния преобразования памяти



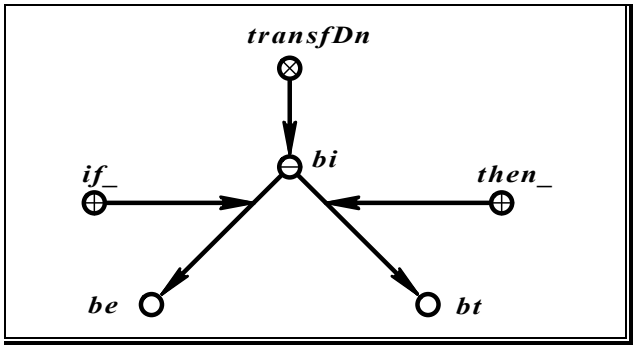
Ниже, на scl-текстах 6.2.6-6.2.8 приведены общие представления отношений для описания состояний перехода в динамической системе.

SCL-текст 6.2.6. Отношение ***implDn***



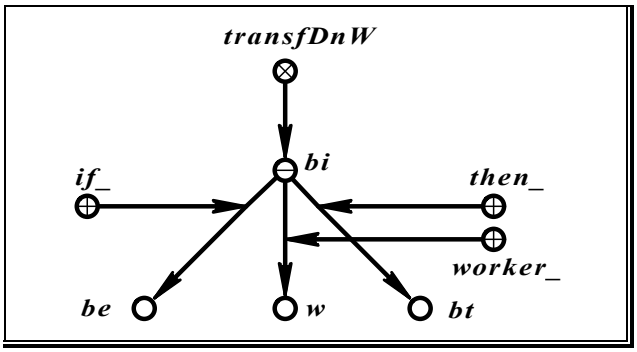
Данная конструкция семантически означает, что для каждого состояния (state-компонента) процесса ***bi*** , которое удовлетворяет условию ***be*** , имеет место также и ***bt*** , т.е. implDn-компонент описывает общее свойство некоторого класса состояний соответствующего процесса.

SCL-текст 6.2.7. Отношение ***transfDn***



Данная конструкция семантически означает, что каждое состояние процесса ***bi*** , удовлетворяющее условию ***be*** , обязательно (независимо ни от чего) преобразуется в следующее за ним состояние, описываемое высказыванием ***bt*** . Нетрудно заметить, что transfDn-высказывания есть способ формального описания всевозможных причинно-следственных связей.

SCL-текст 6.2.8. Отношение ***transfDnW***



Данная конструкция описывает преобразование состояния (ситуации) ***be*** в непосредственно следующее за ним состояние ***bt*** , к которому может привести некий исполнитель ***w*** . Описание условий, необходимых исполнителю ***w*** для выполнения указанного преобразования, также входит в состав высказывания ***be*** .

Перечислим также ряд отношений, заданных на множестве процессов и описывающих различные соотношения между процессами.

Пусть ***bi*** , ***bj*** , ***be*** , ***bt*** , ***ti*** - теории, описывающие нестационарные или квазистационарные процессы. Тогда:

1) конструкция

localBeginStateDn !; ***bi*** ;

означает, что процесс ***bi*** имеет ограничение по моменту своего начала, а конструкция

localBeginStateDn ∀; ***bi*** ;

означает, что для процесса ***bi*** не существует такой точки на оси времени, по отношению к которой момент начала процесса ***bi*** был бы позже, т.е. здесь момент начала процесса ***bi*** стремится к минус бесконечности;

2) конструкция

localEndStateDn !; ***bi*** ;

означает, что процесс ***bi*** имеет ограничение по моменту завершения, а соответственно этому конструкция

localEndStateDn ∀; ***bi*** ;

означает, что процесс ***bi*** длится без конца;

3) конструкция

localBeginStateDn , localEndStateDn !; ***bi*** ;

означает, что процесс ***bi*** полностью локализован во времени;

4) конструкция

localSpaceStateDn !; ***bi*** **;**

означает, что процесс ***bi*** ограничен (локален) в пространстве, т.е. в каждый момент времени, в каждом состоянии процесс ***bi*** не выходит за рамки некоторой пространственной области. При этом в разные моменты времени процесс ***bi*** может находиться в разных местах пространства, т.е. ***bi*** может перемещаться в пространстве;

5) конструкция

bi !; ***state_*** **:** ***bj*** **;**

означает, что процесс ***bj*** является частью (во времени) процесса ***bi*** , т.е. является этапом (стадией, подпроцессом) процесса ***bi*** ;

6) конструкция

bi !; ***beginStateDn_*** **:** ***bj*** **;**

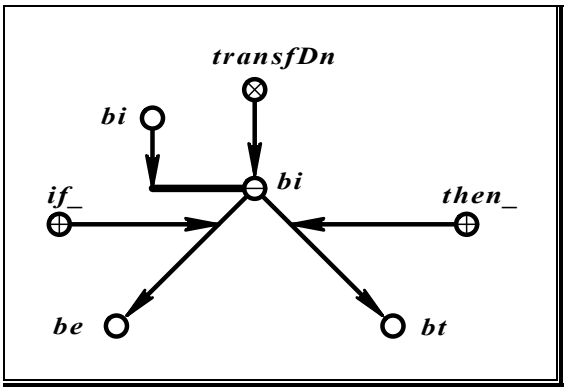
означает, что процесс ***bj*** является начальной стадией (начальным, стартовым этапом, этапом "рождения") процесса ***bi*** , т.е. таким подпроцессом процесса ***bi*** , который начался одновременно с ним, но закончился, естественно, раньше (поскольку это подпроцесс);

7) конструкция

bi !; ***endStateDn_*** **:** ***bj*** **;**

означает, что процесс ***bj*** является конечной стадией (конечным, финишным этапом, этапом завершения) процесса ***bi*** , т.е. таким подпроцессом процесса ***bi*** , который одновременно с ним закончился;

8)



данная конструкция означает то, что процессы ***ti*** , ***be*** и ***bt*** являются подпроцессами процесса ***bi*** . При этом ***ti*** есть процесс преобразования (трансформации) процесса ***be*** в следующий за ним (во времени) процесс ***bt*** , т.е. момент завершения процесса ***be*** совпадает с моментом начала переходного процесса ***ti*** , а момент завершения процесса ***ti*** совпадает с моментом начала (появления, "рождения") процесса ***bt*** ;

9) конструкция

eqBeginStateDn !; □ ***bi*** **,** ***bj*** □ **;**

означает, что процессы (события) ***bi*** и ***bj*** одновременно начались. Здесь ***eqBeginStateDn*** есть свойство "быть одновременно начавшимися процессами", заданное на множестве процессов;

10) конструкция

eqEndStateDn !; □ ***bi*** **,** ***bj*** □ **;**

означает, что процессы ***bi*** и ***bj*** одновременно завершились;

11) конструкция

eqDurationStateDn !; □ ***bi , bj*** □ **;**

означает, что процессы ***bi*** и ***bj*** имеют одинаковую длительность (одинаковое "время жизни"). При этом начаться они могут в разное время;

12) конструкция

comprBeginStateDn !; □ ***bi , grt_ : bj*** □ **;**

означает, что процесс ***bj*** начался позже процесса ***bi*** ;

13) конструкция

comprEndStateDn !; □ ***bi , grt_ : bj*** □ **;**

означает, что процесс ***bj*** кончился позже процесса ***bi*** ;

14) конструкция

comprDurationStateDn !; □ ***bi , grt_ : bj*** □ **;**

означает, что процесс ***bj*** имеет большую длительность, чем процесс ***bi*** ;

15) конструкция

nextTimeStateDn !; □ ***bi , next_ : bj*** □ **;**

означает, что момент завершения процесса ***bi*** совпадает с моментом начала процесса ***bj*** . При этом совсем не обязательно, чтобы ***bi*** и ***bj*** были стадиями (состояниями, подпроцессами) одного и того же процесса.

Кроме того, для процессов (событий) можно ввести такие измеряемые параметры, как отметка времени начала процесса, длительность процесса. Подробнее об этом см. в подразделе 6.1.

Помимо описанных изобразительных средств для обработки знаний, описывающих нестационарные (динамические) предметные области, имеются соответствующие операции. Перечислим основные из них:

1) операции обработки состояний, имеющих некоторое общее свойство. Иначе говоря, это операции реализации implDn-высказываний. Данный класс операций аналогичен операциям реализации продукции в прямом и обратном направлении;
2) операции реализации причинно-следственных связей между состояниями (transfDn-высказываний). С помощью данного класса операций осуществляется переход динамической системы из одного фиксированного состояния в другое;
3) операции реализации состояний, вызванных некоторым исполнителем, т.е. transfDnW-высказываний. Данный класс операций разбивается на три вида. К первому виду относится операция, которая инициируется в случае появления в БЗ некоторого активного исполнителя, который был ранее описан в составе transfDnW-высказывания. В процессе реализации данной операции производится поиск в БЗ всех высказываний, содержащих описание активного исполнителя, а затем из них выбирается то, в одном из компонентов которого описано состояние, в котором находится в текущий момент система. Далее производится переход из текущего состояния в следующее согласно найденному transfDnW-высказыванию. Ко второму виду операций данного класса относится операция, которая инициируется в случае наличия в sc-памяти состояния, описанного в if-компоненте обрабатываемого ею transfDnW-высказывания. В процессе реализации данной операции осуществляется поиск соответствующего активного исполнителя путем формирования запроса. После получения подтверждения от исполнителя осуществляется переход в состояние, описанное в then-компоненте обрабатываемого высказывания. К третьему виду операций реализации состояний, вызванных некоторым исполнителем, относится операция, инициируемая в случае перехода системы в состояние, описанное в then-компоненте обрабатываемого transfDnW-высказывания. Результатом выполнения данной операции является формирование сообщения исполнителю о том, что система перешла в соответствующее состояние;

4) операция поддержки состояний ожидания, т.е. waitState-состояний. Целью данной операции является поиск в памяти sc-конструкций, представляющих собой описание фиксированного состояния, изоморфной конструкции, описанной в обрабатываемом прогнозируемом состоянии ожидания. Данная операция является выполненной успешно в случае успешного поиска указанной конструкции. В обратном случае поиск возобновляется;
5) операции реализации состояний преобразования памяти, т.е. performState-состояний. Данная группа операций на этапе выполнения разбивается на виды, соответствующие тому, какие действия нужно произвести над sc-памятью. Каждый из видов является аналогом scl-оператора в обобщенном виде. Первая из операций данного класса выполняет генерацию в sc-памяти sc-элементов, помеченных в описании performState-состояния атрибутом ***generate_***. Вторая осуществляет удаление sc-элементов, помеченных атрибутом ***delete_***. Результатом выполнения третьей операции данного класса является загрузка в sc-память содержимого sc-узла, помеченного атрибутом ***input_*** в составе performState-состояния;
6) операции поддержки различных соотношений между состояниями. Данный класс scl-операций разбивается на виды, каждый из которых осуществляет обработку отношений, задаваемых с помощью следующих ключевых узлов: ***localBeginStateDn***, ***localEndStateDn***, ***localSpaceStateDn***, ***beginStateDn_***, ***endStateDn_***, ***eqBeginStateDn***, ***eqEndStateDn***, ***eqDurationStateDn***, ***comprBeginStateDn***, ***comprEndStateDn***, ***comprDurationStateDn***, ***nextTimeStateDn***.

Каждая операция, таким образом, в рамках динамической системы представляет собой некий процесс по обработке соответствующего типа знаний.

6.3. Описание целей в графодинамических ассоциативных машинах

Ключевые понятия и идентификаторы ключевых узлов: цель, **задание**, **команда**, **подцель**, **информационная цель**, **поведенческая цель**, **задача**.

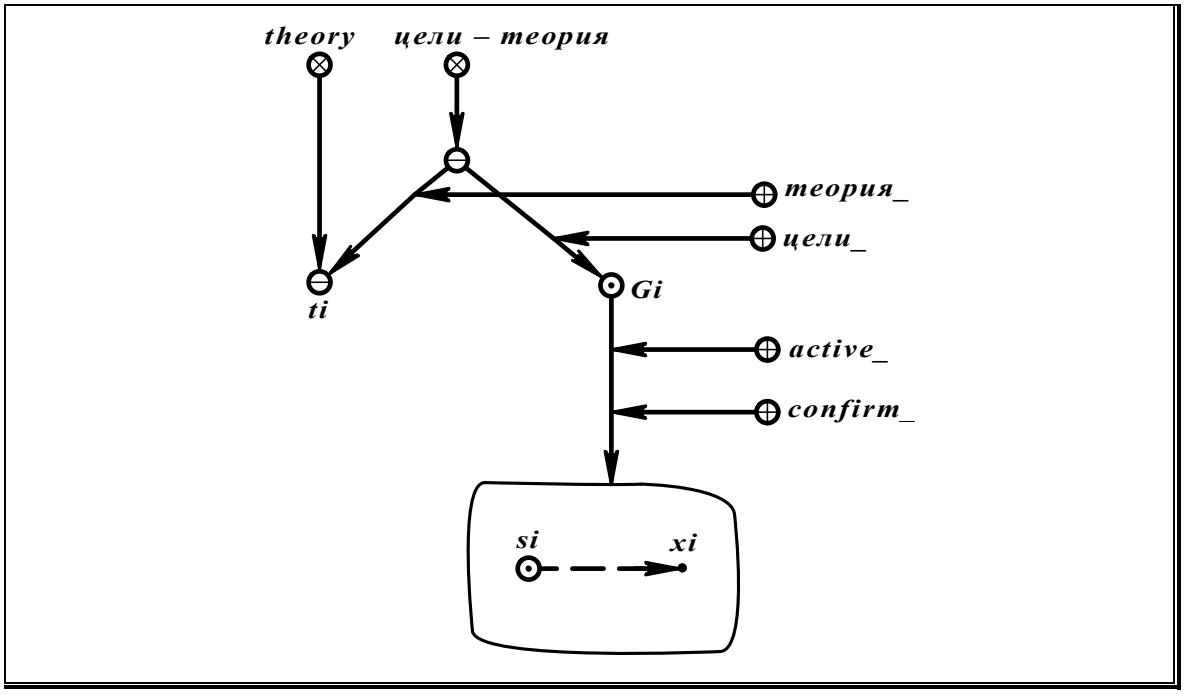
Важнейшей частью любого языка представления знаний являются средства явного описания целей, поскольку по принципу инициирования элементарных информационных процессов машины переработки знаний относятся к классу машин, управляемых потоком целей (см. подраздел 1.3.).

Цель (задание) – это либо желаемое и подлежащее достижению состояние информационной конструкции, хранимой в памяти машины переработки информации (такие цели будем называть информационными), либо желаемое и подлежащее достижению состояние внешней среды (такие цели будем называть поведенческими или внешними). Частным видом описания информационных целей можно считать команды – информационные операторы или программы, которые представляют собой описание подлежащих выполнению действий, направленных на переработку хранимых в памяти информационных конструкций. Действительно, в информационном операторе или информационной программе пусть и не явно, но все же содержится указание на будущее (целевое) состояние хранимой информационной конструкции, к которому машина переработки информации стремится (т.е. указание на состояние, которое является результатом реализации оператора или программы). Напомним при этом, что реализация процедурных программ сводится к реализации операторов, входящих в эти программы. Совершенно аналогичным образом частным видом описания поведенческих (внешних) целей можно считать поведенческие операторы или программы, направленные на преобразование внешней среды. Принципиальная разница между оператором, программой и описанием цели, которое не является ни оператором, ни программой, заключается в том, что оператор и программа есть описание цели, совмещенное с явным указанием метода (способа) достижения этой цели. Тогда, как описание цели, не являющееся ни оператором, ни программой, не содержит в себе явного указания на метод (способ) достижения такой цели. Этот метод будет зависеть от того, какой контекст имеет указанная цель. Анализ этого контекста и выбор соответствующего метода достижения цели как раз и составляет основу всех моделей переработки знаний. Такие цели будем называть информационными заданиями (запросами, вопросами, описаниями информационных потребностей). В данном подразделе будут рассмотрены описания информационных целей, не являющиеся ни операторами, ни программами. Принципы представления операторов и программ средствами языка SC рассматриваются в [411] (***ПрогрВАМ-2001кн***). Таким образом, будем различать цели на задания и команды.

Для представления информационных заданий в язык SCL вводятся специальные ключевые узлы (***goal***, ***confirm_***, ***deny_***), которые непосредственно определяют целевые состояния самого общего вида. Все многообразие целей задается различными контекстами указанных целей общего вида. На scl-тексте 6.3.1 показан самый общий вид информационных заданий, представляемых с помощью ключевого узла ***confirm_***. Выполнение такого вида задания предполагает довольно сложный анализ и изменение всей окрестности (всего контекста) соответствующей всем

sc-переменным включённым в задание. При этом характер и метод (алгоритм) такого изменения заранее не известен и определяется структурой контекста sc-переменных.

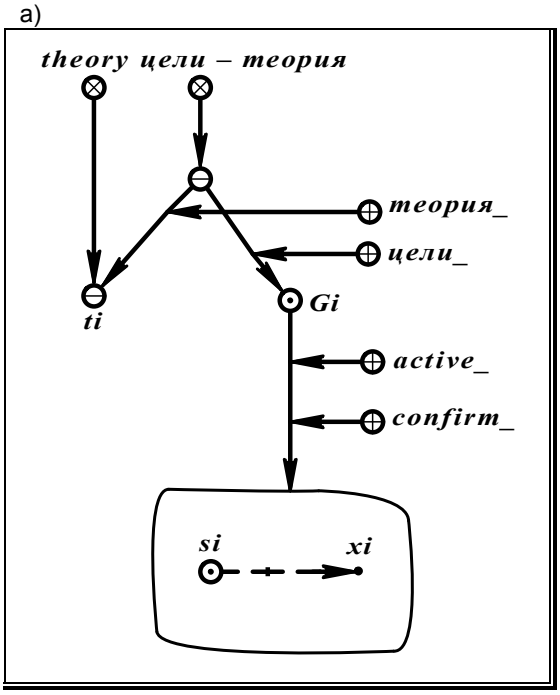
SCL-текст 6.3.1. Задание на преобразование текущего состояния обрабатываемой scl-теории ***ti*** таким образом, чтобы было подтверждено наличие константной позитивной sc-дуги, выходящей из узла ***si*** и входящей в элемент ***xi*** (в зависимости от контекста, это задание может подразумевать превращение соответствующей негативной или нечеткой sc-дуги в позитивную). ***Gi*** – множество целей формальной теории ***ti***



Используемые в абстрактной scl-машине частные виды информационных заданий, описываемых с помощью ключевого узла ***confirm_*** , показаны на scl-текстах 6.3.3 – 6.3.15.

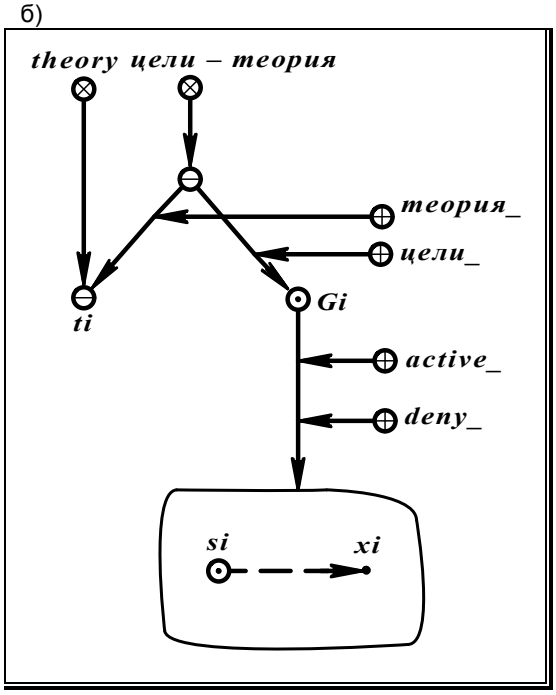
SCL-текст 6.3.2. Задание на преобразование текущего состояния обрабатываемой scl-теории ***ti*** таким образом, чтобы было опровергнуто наличие константной позитивной sc-дуги, выходящей из узла ***si*** и входящей в элемент ***xi*** (в зависимости от контекста, это задание может подразумевать превращение соответствующей позитивной или нечеткой sc-дуги в негативную)

а)

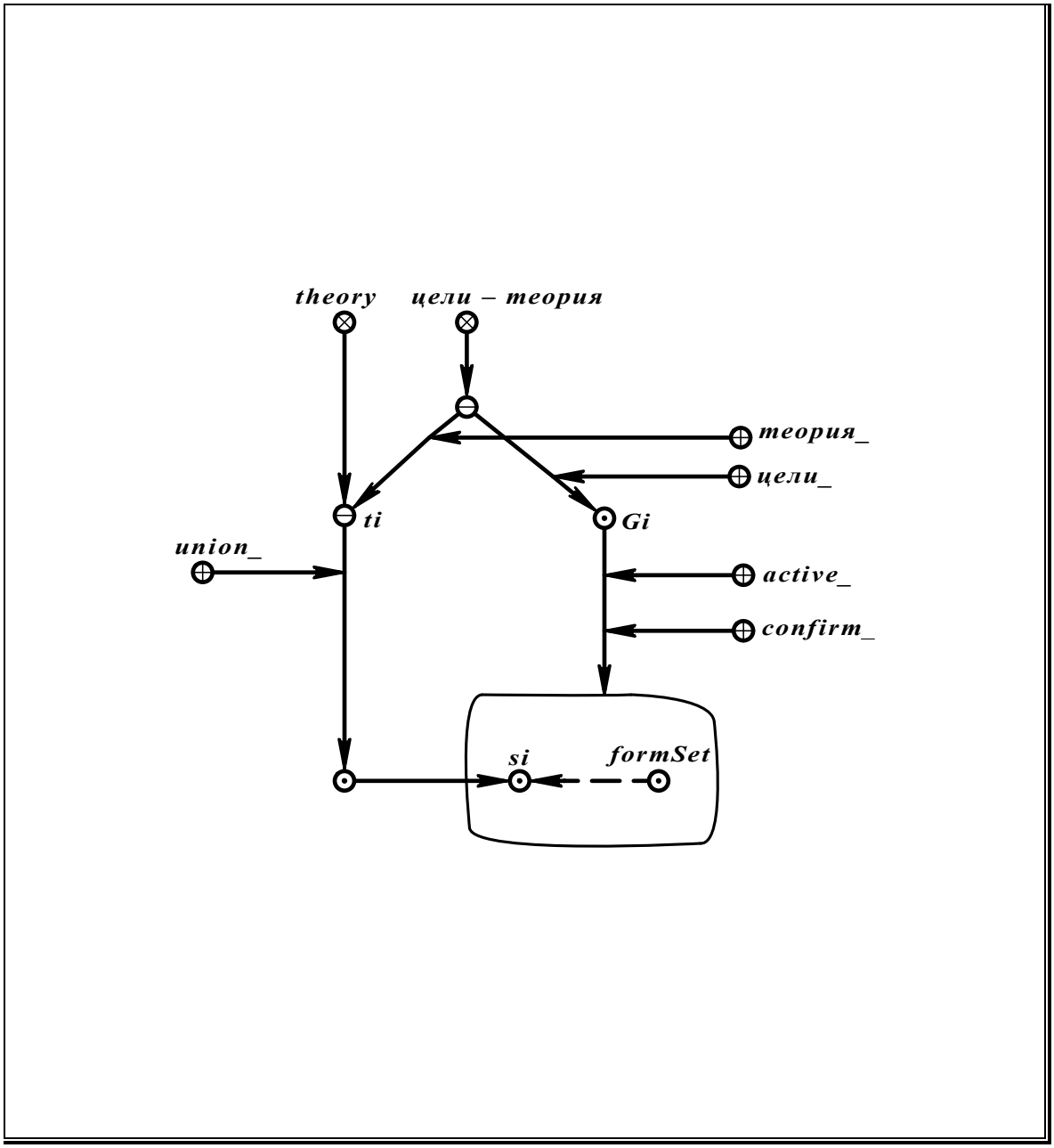


<#;

б)



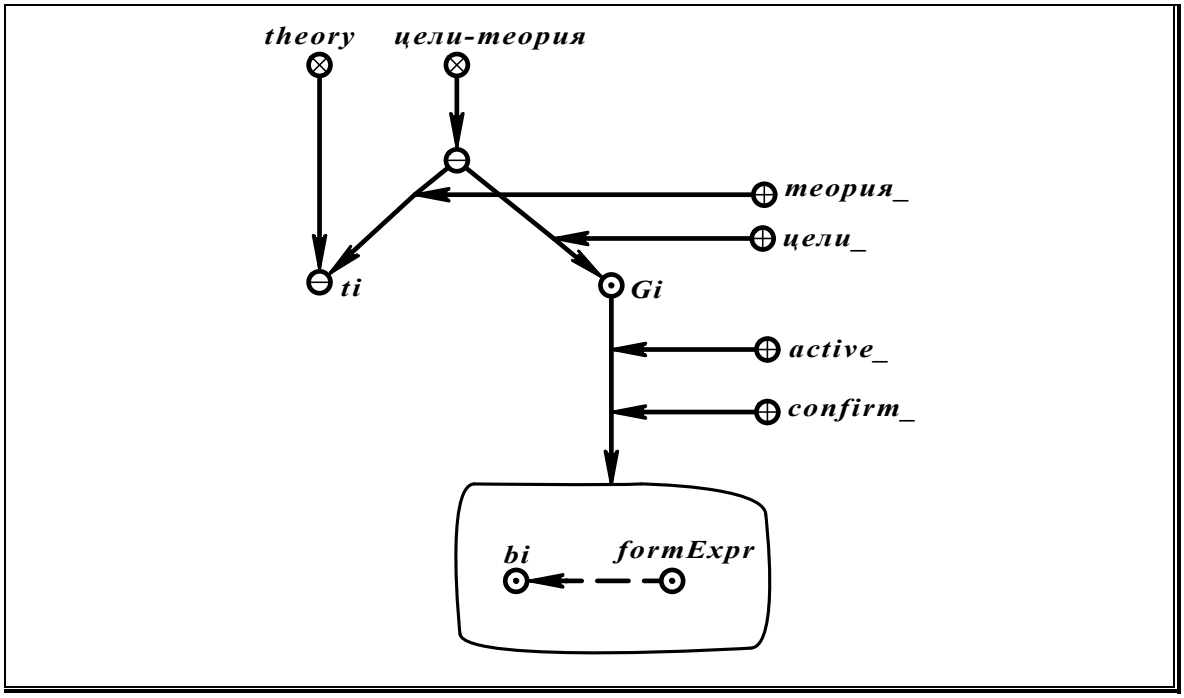
SCL-текст 6.3.3. Задание на построение полного перечня всех элементов множества ***si***



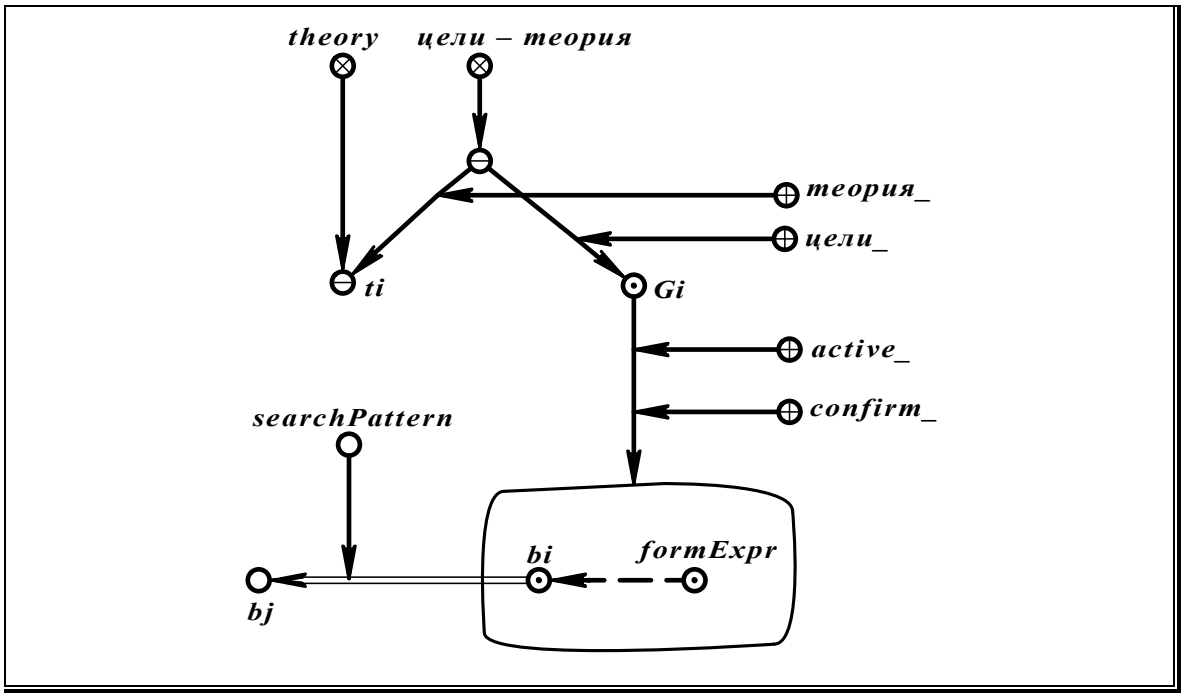
На scl-тексте 6.3.3 приведен пример задания на построение полного перечня всех элементов множества ***si***. Результатом выполнения этого задания является построение всех константных sc-дуг, выходящих их sc-узла ***si***. При этом входить указанные sc-дуги могут как в константные, так и в переменные sc-элементы. Заметим также, что изначально из узла ***formSet*** константная дуга может как выходить – нечёткая, либо негативная, так и вообще отсутствовать.

На scl-тексте 6.3.4 приведён пример задания на формирование логической формулы ***bj***, входящей в состав (подчиненной) scl-теории ***ti***. Напомним, что все константные sc-узлы, обозначающие scl-формулы, входящие в состав scl-теории, а также все константные sc-дуги, инцидентные этим sc-узлам, по умолчанию считаются константами указанной scl-теории. Сформировать scl-формулу – это значит сформировать не только компоненты этой scl-формулы, но и все остальные scl-формулы, входящие в её состав, вплоть до атомарных.

SCL-текст 6.3.4. Задание на формирование scl-формулы ***bj***, входящего в состав scl-теории ***ti***



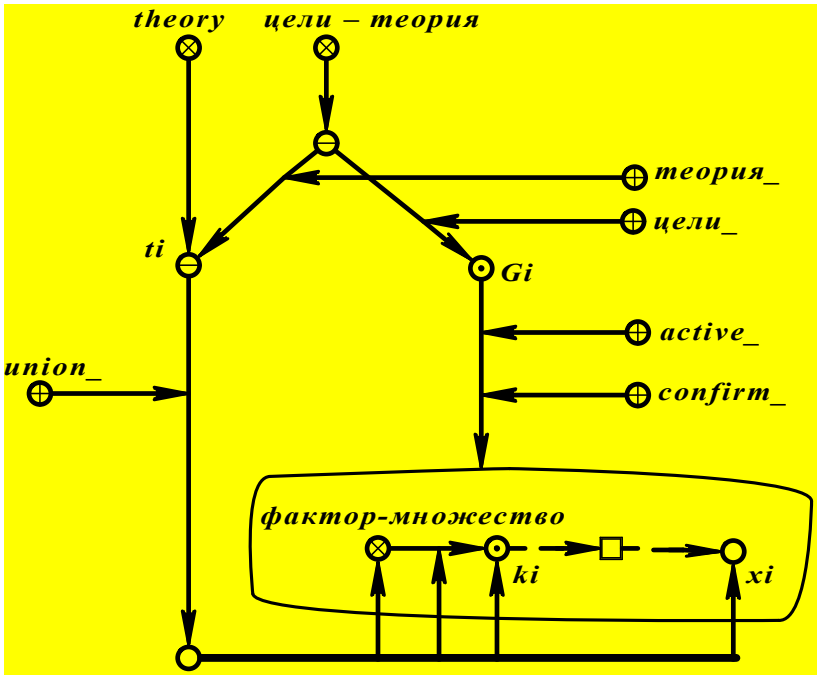
SCL-текст 6.3.5. Общий вид задания на формирование или информационный поиск



На scl-тексте 6.3.5 приведен общий вид задания на формирование или информационный поиск одного из высказываний (***bi***), удовлетворяющих образцу поиска, заданному высказыванием ***bj***. На высказывание ***bj*** должен неявно навешиваться квантор существования. Формируемое высказывание ***bi*** отличается от высказывания ***bj*** только тем, что в нем все переменные, связываемые в высказывание ***bj*** квантором существования, заменяются на константы теории ***ti***. Это задание является частным по отношению к заданию на формирование scl-формулы. Граф, информационная конструкция, описывающая частную цель всегда является подграфом графа, описывающего цель более общего вида, что можно видеть из рисунков 6.3.5 и 6.3.4.

На scl-тексте 6.3.6 приведен пример задания на классификацию (распознавание), т.е. на определение того, к какому классу из заданного семейства ki относится заданный объект xi. Как можно видеть это задание не является частным по отношению к заданию, показанному на scl-тексте 6.3.4.

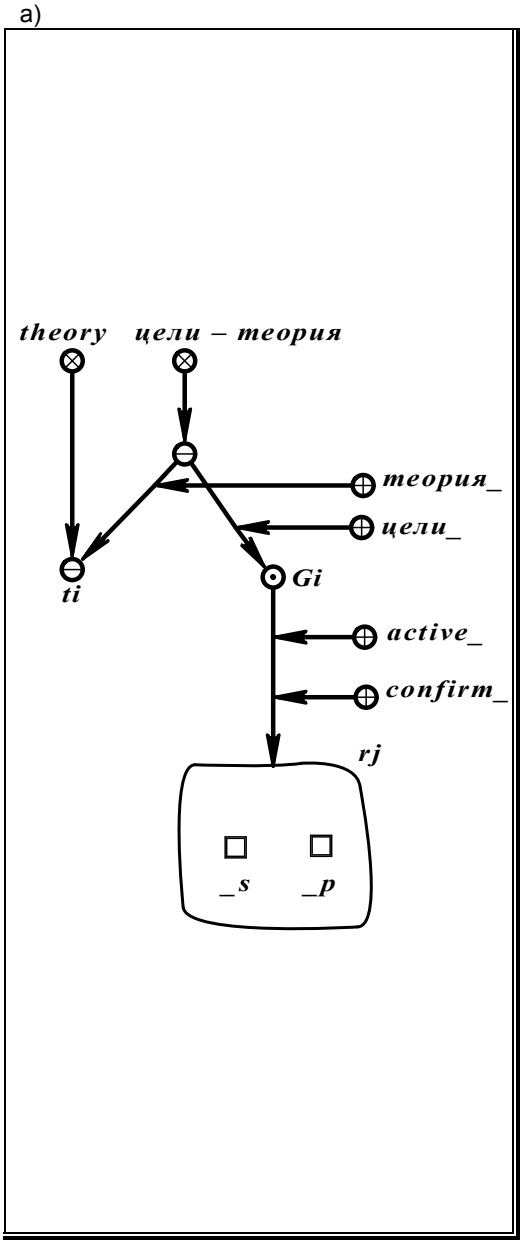
SCL-текст 6.3.6. Задание на классификацию



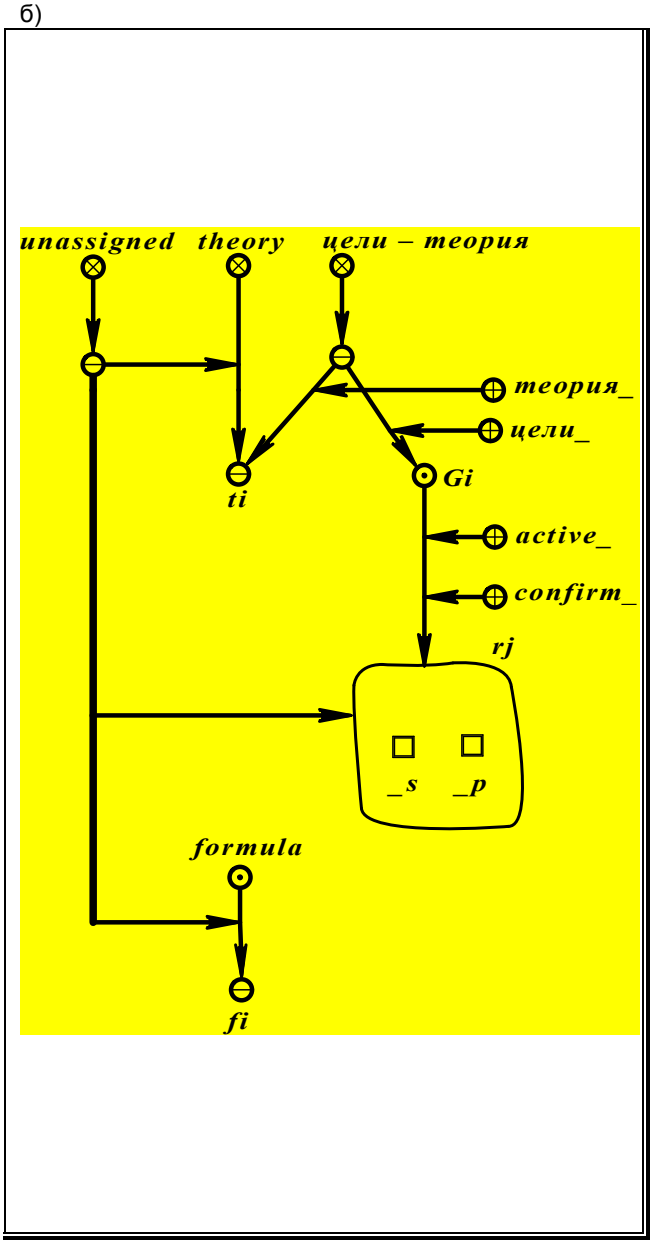
На scl-тексте 6.3.7 приведен пример задания на логический вывод или на информационный поиск всех фактографических высказываний, удовлетворяющих заданному образцу поиска. Вариант б) – это частный вид такого задания, по отношению к заданию варианта а).

SCL-текст 6.3.7. Задание на логический вывод формулы ***rj***, удовлетворяющей ограничениям заданным формулой ***fi***:

а)

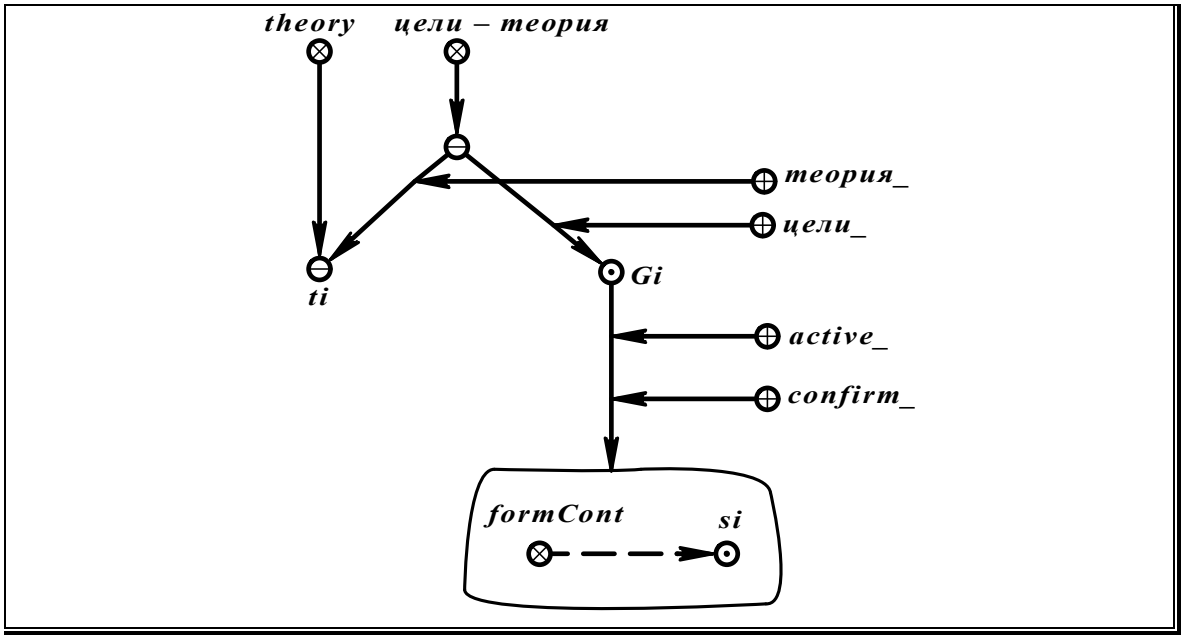


б)



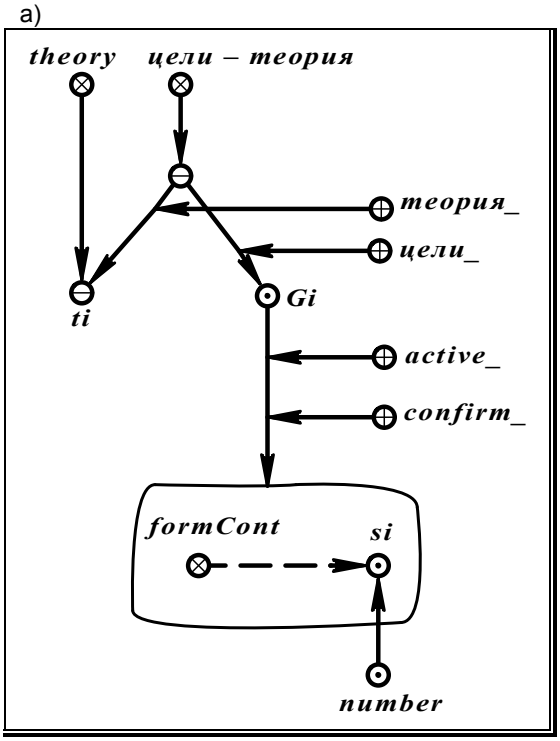
На scl-тексте 6.3.9 а) приведен пример задания на вычисление числа, обозначенного sc-узлом ***si*** . В результате выполнения этого задания sc-узел ***si*** может быть склеен с другим числовым sc-узлом, имеющим такое же фиксированное содержимое. Числовые sc-узлы, имеющие одинаковое и фиксированное содержимое, семантически эквивалентны. Данное задание является частным по отношению к заданию на формирование содержимого (см. scl-текст 6.3.8)

SCL-текст 6.3.8. Задание на формирование содержимого sc-узла ***si***

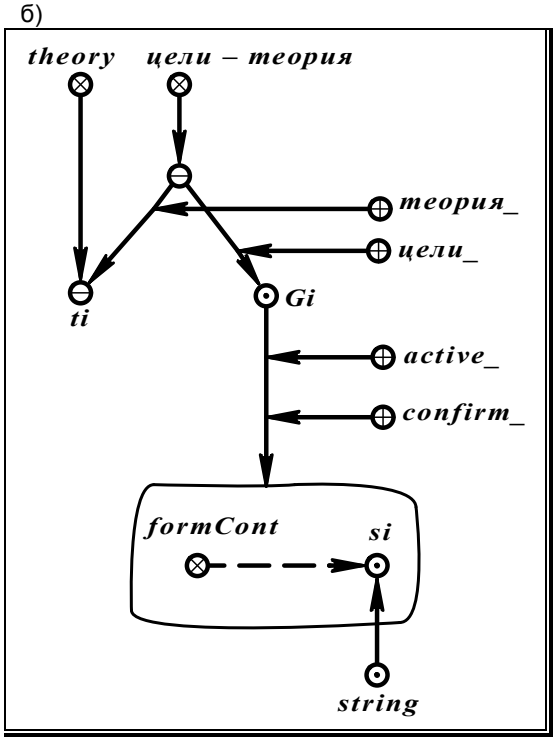


SCL-текст 6.3.9. Задание на вычисление содержимого определённого типа (числа, строки) узла ***si***

а)

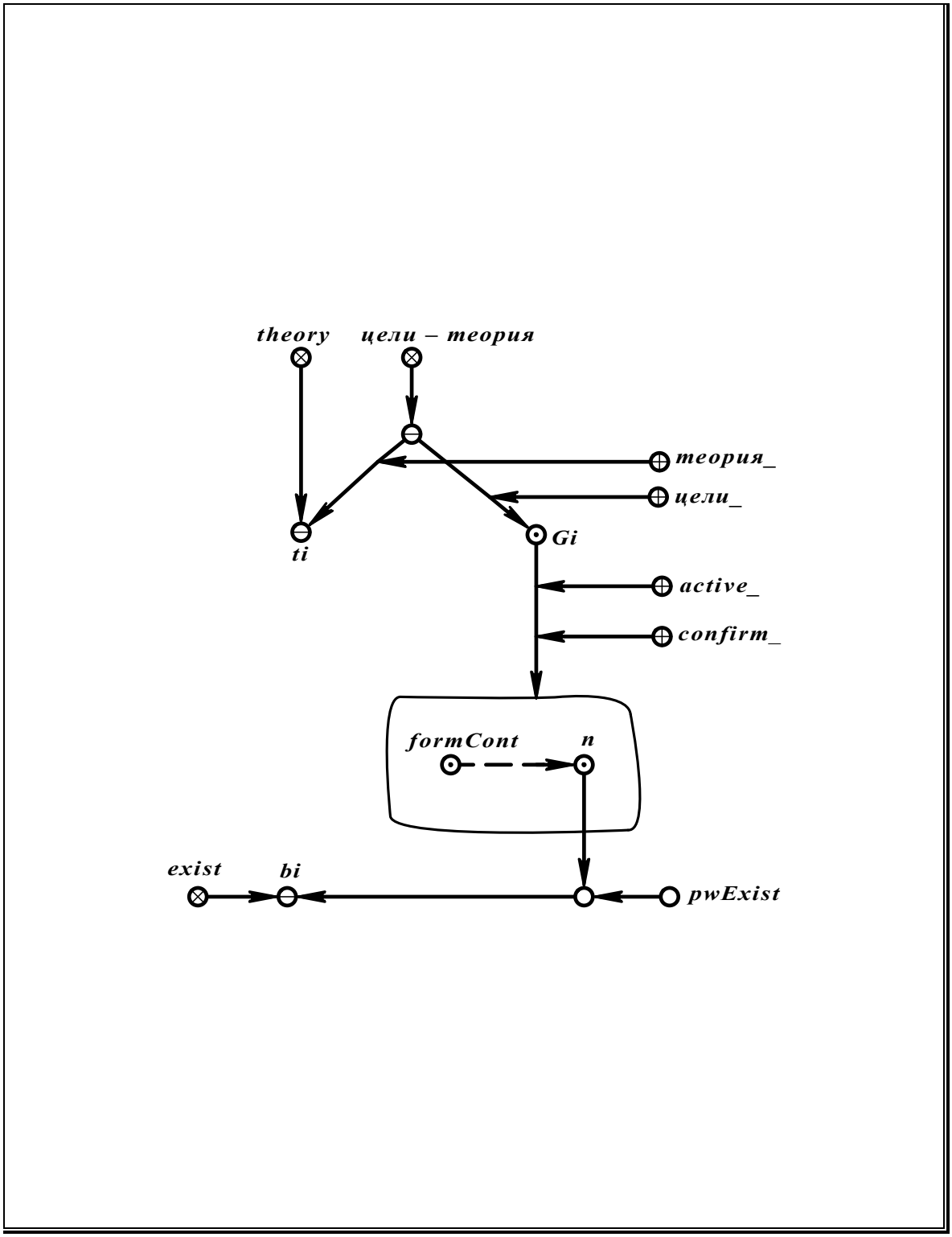


б)

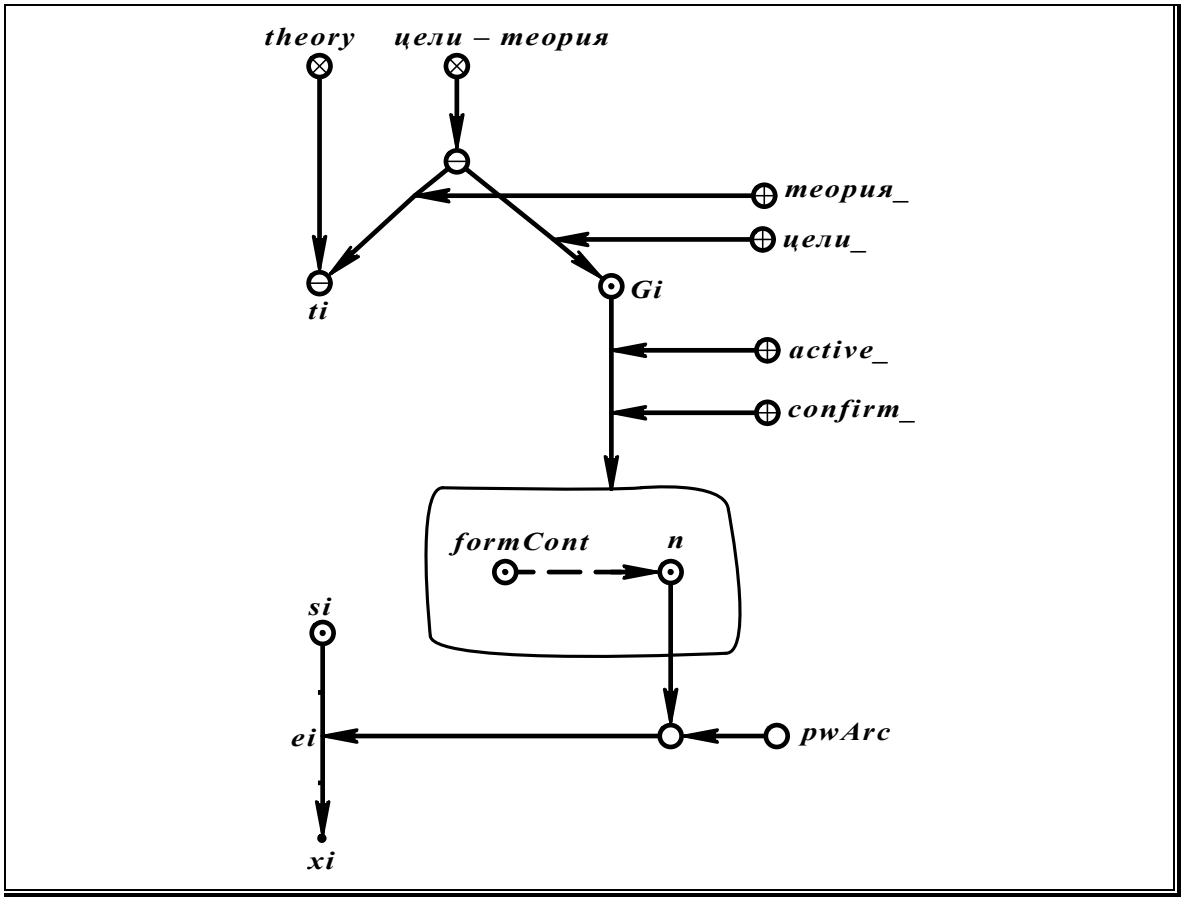


На scl-тексте 6.3.10 приведен пример задания на определение количества всех высказываний, удовлетворяющих формуле ***bj*** в рамках scl-теории ***ti***. В частности, это может быть заданием на перечисление объектов, удовлетворяющих заданным свойствам. Это задание является частным по отношению к заданию на вычисление числа (см. scl-текст 6.3.9).

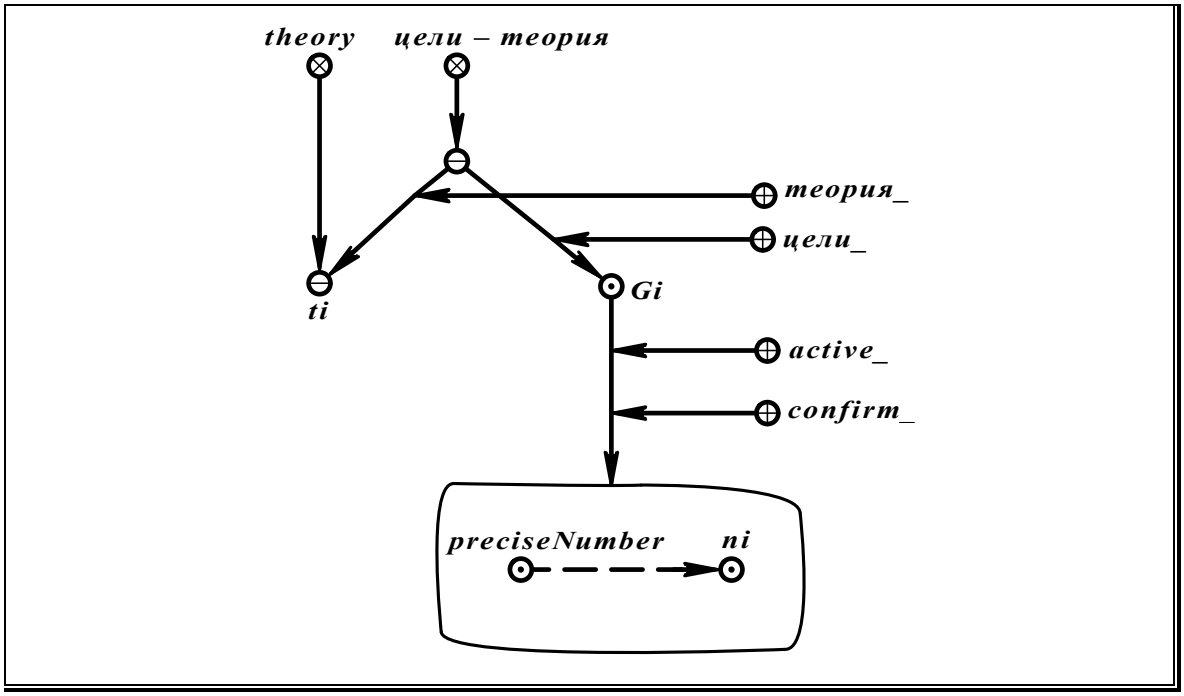
SCL-текст 6.3.10. Задание на определение количества всех (фактографических) высказываний, удовлетворяющих формуле ***bj***



SCL-текст 6.3.11. Задание на определение степени четкости нечеткой sc-дуги *ei*

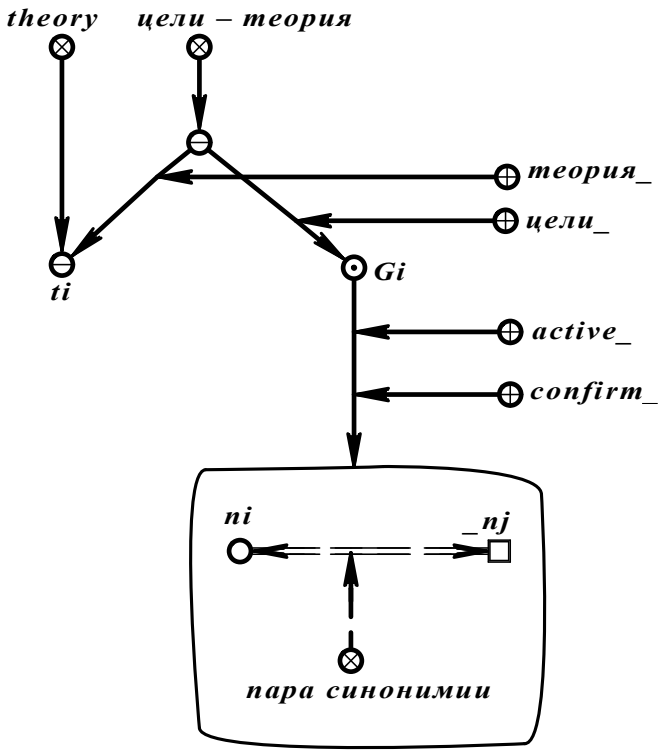


SCL-текст 6.3.12. Задание на повышение точности числа *ni*

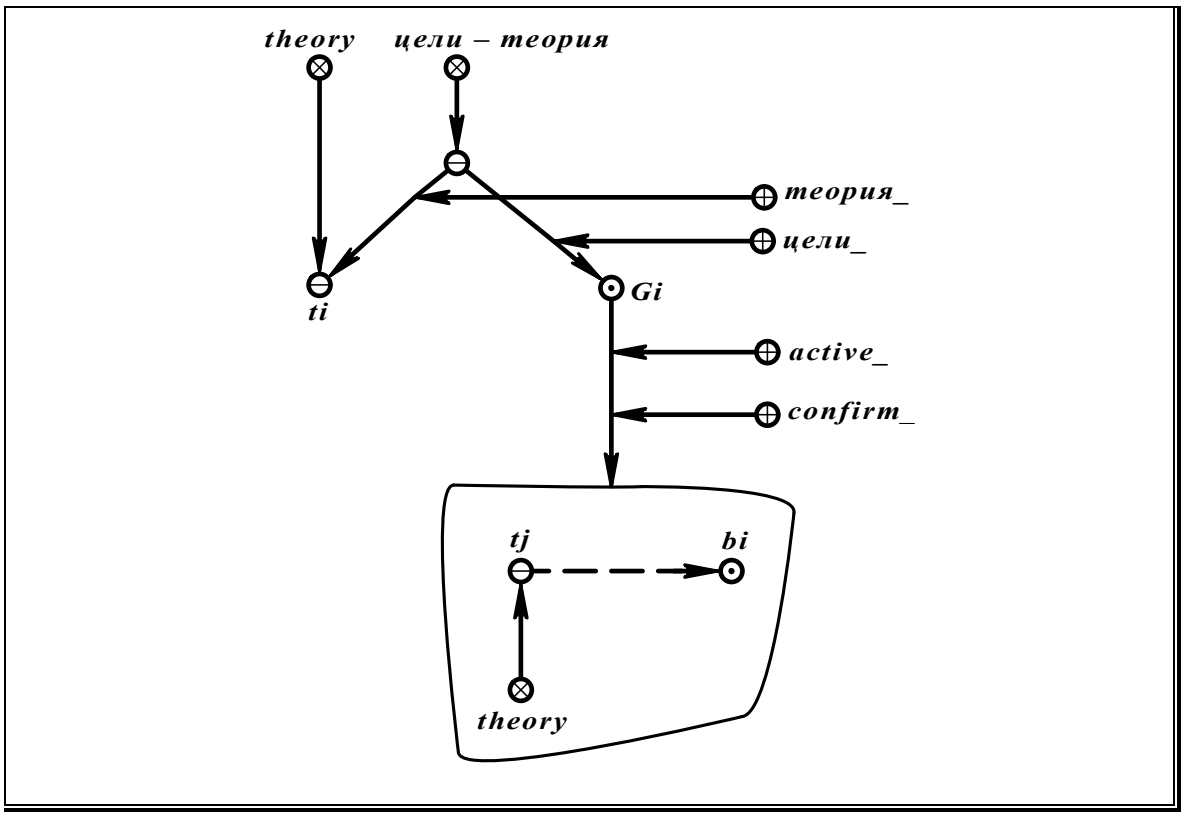


На scl-тексте 6.3.13 приведен пример задания на поиск среди всех констант scl-теории ***ti*** (т.е. среди констант, явно перечисленных в текущем состоянии scl-теории) такого sc-узла, который семантически эквивалентен sc-узлу ***ni***. Результатом выполнения этого задания является склеивание sc-узла ***ni*** с найденным sc-узлом.

SCL-текст 6.3.13. Задание на поиск семантически эквивалентных sc-элементов



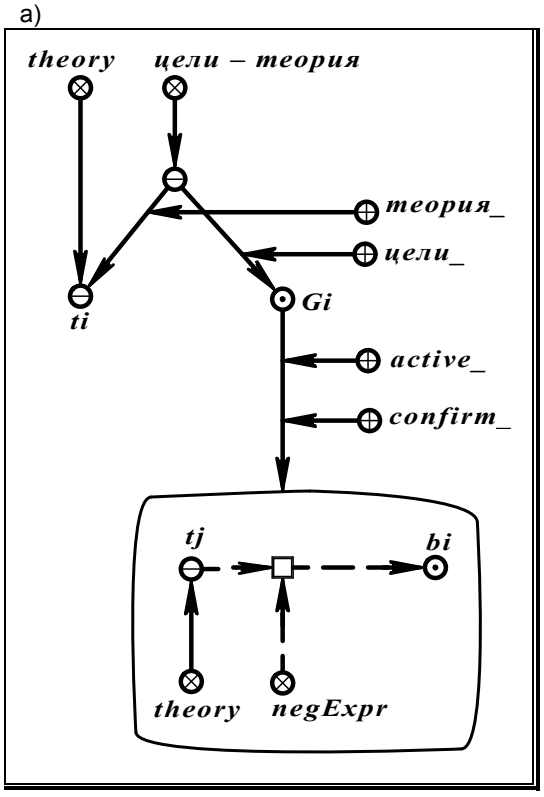
SCL-текст 6.3.14. Задание на установление истинности высказывания ***bi*** теории ***ti***



На scl-тексте 6.3.14 приведен пример задания на установление истинности scl-высказывания (утверждения) ***bi*** в рамках scl-теории ***ti***. SCL-высказывание ***bi*** может быть высказыванием любого вида.

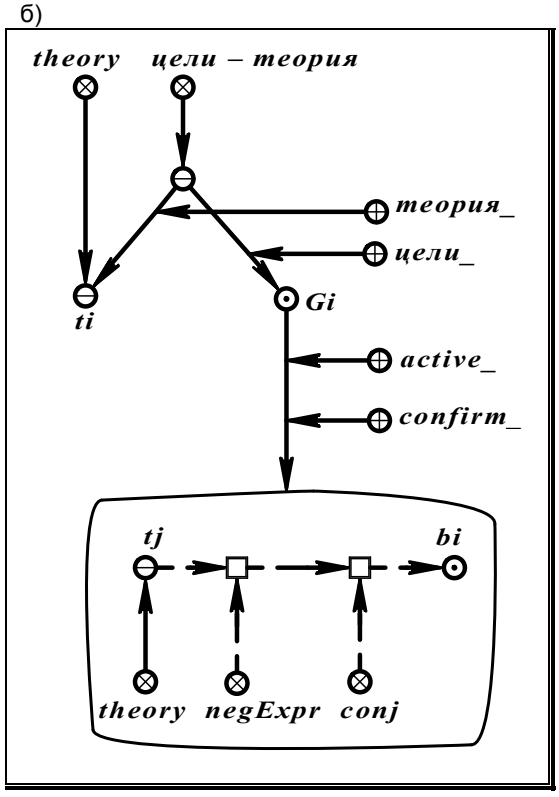
SCL-текст 6.3.15. Задание на установление ложности высказывания ***bi*** теории ***ti***

а)



<#;

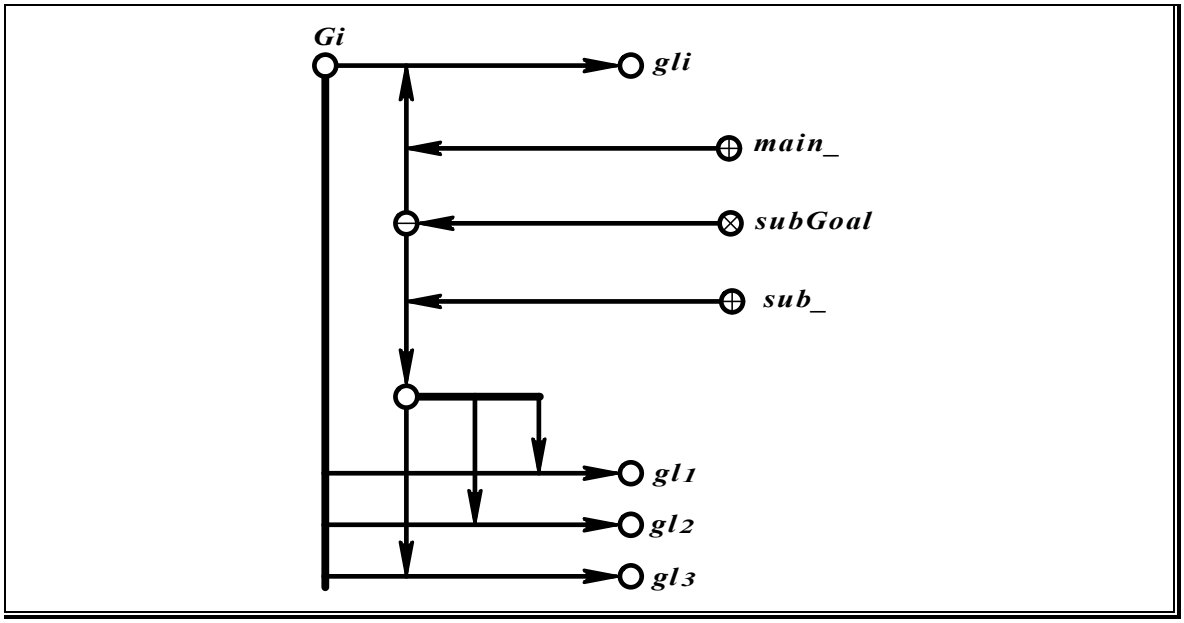
б)



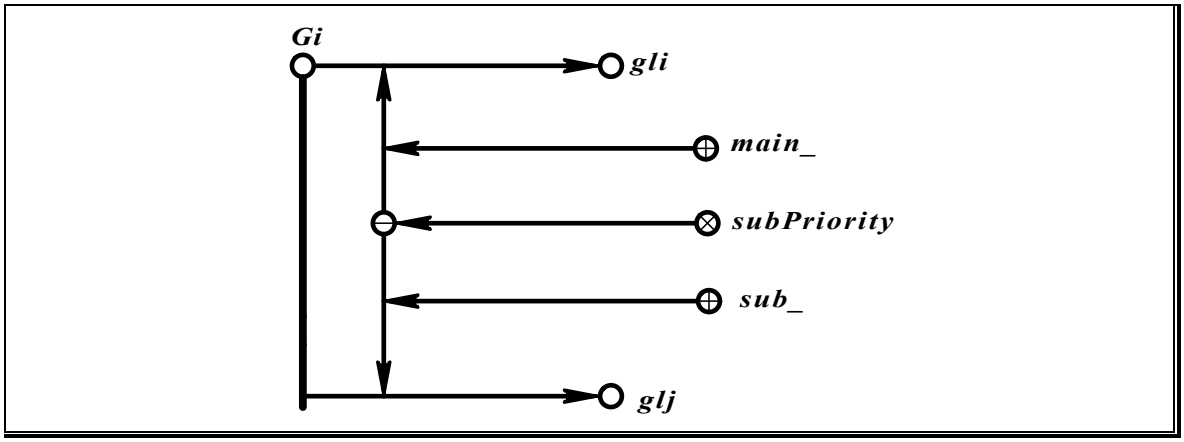
На множестве целей задан целый ряд отношений, наиболее важными из которых являются отношение, связывающее (основные) цели с их И-подцелями (И-подцели – набор целей, достижение каждой из которого и только достижение каждой из которого гарантирует достижение основной цели), и отношение, упорядочивающее множество целей по их приоритету (важности). Формально областью определения этих отношений будем считать константные позитивные sc-дуги, принадлежащие множествам ***confirm_*** , ***deny_*** . Отношение, связывающее цели с их И-подцелями, обозначается ключевым узлом ***subGoal*** , является асимметричным отношением нефиксированной арности и использует два атрибута (***main_*** и ***sub_***). Кортеж отношения ***subGoal*** связывает некоторую цель, указываемую под атрибутом ***main_*** , со всеми ее И-подцелями, каждая из которых отмечается атрибутом ***sub_*** . Смысл И-подцелей заключается в том, что после достижения всех И-подцелей достижение исходной цели гарантируется с помощью метода, известного scl-машине. Подчеркнем, что каждая цель может быть сведена к своим И-подцелям в общем случае несколькими способами. Стоит отличать понятие подцели от понятия частной цели.

Отношение, упорядочивающее множество целей по их приоритету, обозначается ключевым узлом ***goalPriority*** , является асимметричным бинарным отношением и использует два атрибута (***main_*** и ***sub_***). Кортеж отношения ***goalPriority*** сравнивает две цели, одна из которых (указываемая под атрибутом ***main_***) считается более приоритетной по отношению к другой цели.

SCL-текст 6.3.16. Пример связки отношения ***subGoal***



SCL-текст 6.3.17. Пример связки отношения ***goalPriority***



Каждой цели ставится в соответствие некий субъект, являющийся автором (постановщиком) этой цели. В частном случае автором цели может быть сама интеллектуальная система. Если цель поставлена

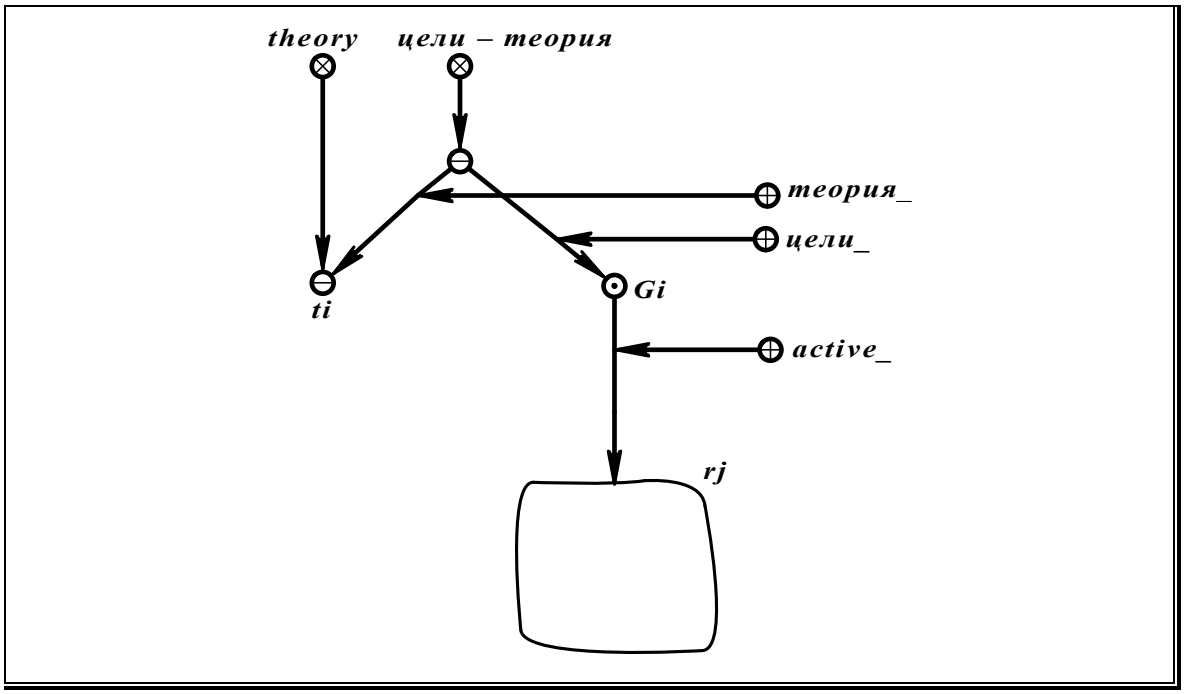
другим (внешним) субъектом, то интеллектуальная система должна определить корректность (возможность достижения) этой цели и, в случае если цель поведенческая, соответствие ее принятым нормам (правилам) поведения интеллектуальной системы во внешней среде. После всего этого интеллектуальная система может принять или не принять к исполнению цель, поставленную внешним субъектом. Последнее означает, что указанная цель становится также собственной целью интеллектуальной системы.

Цели могут быть как инициированными, т.е. подлежащими достижению (выполнению) в текущий период времени, так и неинициированными (в частности, достигнутыми). Инициированные цели в языке SCL дополнительно отмечаются специальным атрибутом ***active_*** , см. scl-текст 6.3.18. Достигнутые цели помечаются соответствующим атрибутом ***denied_*** или ***confirmed_*** соответственно тому: был ли запрос ***rj*** подтверждён, либо опровергнут.

Более подробное рассмотрение средств описания целей в языке SCL приведено в работе [151] (***Голенков В.В..1995пр-Предс3РВ***).

Задача, решаемая интеллектуальной системой (задачная ситуация) определяется 1) некоторым исходно заданным конструктивным объектом и 2) описанием некоторой цели – обобщенным описанием свойств требуемого (результирующего) конструктивного объекта. Задача может быть не решаема (не корректна) либо из-за неполноты исходных данных, либо из-за противоречия между исходными данными и целью. Исходными данными (исходным конструктивным объектом) для задачи, формулируемой в рамках базы знаний, может быть, в частности, все текущее состояние указанной базы знаний. Задачи являются неотъемлемыми компонентами баз знаний и процесса функционирования интеллектуальной системы в целом, так как функционирование интеллектуальной системы трактуется как совокупность взаимодействующих процессов, каждый из которых направлен на решение некоторой конкретной задачи. Каждая возникающая в базе знаний задача инициирует в общем случае несколько параллельных процессов, пытающихся решить эту задачу разными способами.

SCL-текст 6.3.18. Общий вид инициированных заданий



6.4. Гипертекстовые семантические сети

Ключевые понятия и идентификаторы ключевых узлов: гипертекстовая семантическая сеть, ***информационная конструкция***, ***текст***, ***изображение***, ***видео-информация***, ***аудио-информация***, ***семантическая эквивалентность текстов***.

Язык SC может быть использован для построения гипертекстовых семантических сетей. Такие семантические сети включают разнородные информационные конструкции, для связи которых используются метаязыковые и ассоциативные возможности языка SC. Для просмотра гипертекстовых семантических сетей необходима соответствующая навигационно-поисковая графодинамическая ассоциативная машина (см. раздел 7). Для управления методами вывода различных информационных конструкций в зависимости от их класса вводятся понятие стиля отображения (воспроизведения) и специальные отношения между стилем воспроизведения и воспроизводимыми информационными конструкциями.

Пусть дано:
- множество информационных конструкций самого различного вида (тексты, изображения, видеоинформация, аудиоинформация);
- множество знаков, обозначающих самые различные объекты (конкретные предметы некоторой предметной области, конкретные информационные конструкции, конкретные множества, связи, понятия, отношения). В языке SC такие знаки в основном представлены sc-узлами и реже sc-связками (дугами, ребрами);
- множество идентификаторов (имен), которые взаимно однозначно соответствуют множеству вводимых знаков и являются строковым (линейно-символьным) вариантом изображения знаков.

При этом будем считать, что:
- sc-узел, являющийся знаком некоторой информационной конструкции, **содержит** обозначаемую им информационную конструкцию (другими словами, информационная конструкция считается **содержимым** того sc-узла, который её обозначает);
- информационная конструкция (в частности, текстовая информационная конструкция) может включать в себя идентификаторы (имена) некоторых знаков, представленных sc-узлами. Это трактуется как ссылка на соответствующий знак (sc-узел);
- типология информационных конструкций, соотношения между ними, а также их синтаксическая структура и семантика, описываются в виде sc-конструкций с помощью целого ряда вводимых понятий и отношений.

Гипертекстовой семантической сетью будем называть некоторое заданное множество информационных конструкций вместе с sc-конструкцией, являющейся надстройкой над множеством sc-узлов, обозначающих указанные информационные конструкции, и описывающей типологию этих информационных конструкций, соотношения между ними, а также их синтаксическую структуру и семантику. Гипертекстовую семантическую сеть можно также называть:
- семантически структурированным гипертекстом;
- результатом интеграции гипертекстовых технологий и технологий, основанных на семантических сетях;
- семантически структурированной гипертекстовой мультимедийной базой знаний.

В языке SC для информационных конструкций, которые не является текстами языка SC, используется специальный способ их представления и хранения в виде содержимого предметных sc-узлов. Информационные конструкции образуют иерархию, так как всегда можно построить новую информационную конструкцию, которая является объединением других информационных конструкций.

Примечание. Тот sc-узел, который обозначает некоторую информационную конструкцию, вовсе не обязан явно “хранить” эту информационную конструкцию в качестве своего содержимого. Эта информационная конструкция может быть неизвестна (не сформирована). Эта информационная конструкция может быть разбита на фрагменты, каждый из которых представлен явно, и, следовательно нет никакой необходимости явно представлять и хранить всю исходную информационную конструкцию.

Рассмотрим некоторые понятия, определяющие типологию информационных конструкций. Для каждого такого понятия вводится sc-узел, обозначающий соответствующий тип информационных конструкций:

информационная конструкция
- ***текст*** (дискретная информационная конструкция)
  - ***ея-текст*** (текст естественного языка)

- ***официальный документ*** (административный документ)
  - ***приказ***
  - ***служебная записка***
  - ***заявление***
  - ***закон***
  - ***договор***
  - ***протокол заседания***
- ***художественный текст***
  - ***рассказ***
  - ***повесть***
  - ***роман***
  - ***стихотворение***
  - ***поэма***
- ***научно-технический документ***
  - ***научно-технический отчет*** (отчет о научно-исследовательской работе)
  - ***техническое задание***
  - ***научная монография***
  - ***научная статья***
- ***учебно-методический документ***
  - ***учебный план специальности***
  - ***квалификационная характеристика специальности***
  - ***программа учебной дисциплины***
  - ***учебное пособие***
  - ***учебник***
  - ***методическое пособие***
- ***текст формального языка***
  - ***нотный текст***
  - ***scg-текст*** (текст языка SCg)
  - ***scs-текст*** (текст языка SCs)
  - ***текст логического языка***
    - ***sclg-текст*** (текст языка SCLg)
    - ***scls-текст*** (текст языка SCLs)
  - ***текст языка программирования***
  - ***текст языка представления знаний***
  - ***текст языка запросов***
  - ***текст фактографических языков***
    - ***scbg-текст*** (текст языка SCBg)
    - ***scbs-текст*** (текст языка SCBs)
- ***изображение***
  - ***чертеж***
  - ***карта***
  - ***фотография***
  - ***рисунок***
- ***аудиоинформация***
  - ***запись музыкального произведения***
  - ***запись речевого сообщения***
  - ***аудиозапись интервью***
  - ***аудиозапись лекции***
  - ***аудиозапись выступления***
- ***видеоинформация***
- ***видеоаудиоинформация***
  - ***художественный фильм***
  - ***документальный фильм***
  - ***видеозапись интервью***
  - ***видеозапись лекции***
  - ***видеозапись выступления***

- ***мультфильм***

текст
- ***линейный текст*** (текст линейного языка)
- ***нелинейный текст*** (текст графового языка)

ея-текст
- ***русский текст*** (текст русского языка)
- ***английский текст*** (текст английского языка)
- ***немецкий текст*** (текст немецкого языка)

ея-текст
- ***ея-определение***
- ***ея-пояснение*** (нестрогое определение)
- ***ея-утверждение*** (ея-описание некоторой закономерности)
- ***фактографический ея-текст***
- ***ея-комментарий***
- ***ея-вопрос*** (ея-формулировка информационной цели)
- ***ея-формулировка поведенческой цели***
- ***ея-формулировка задачи***
- ***ея-запись информационной программы*** (программы обработки информации)
- ***ея-запись поведенческой программы***
- ***ея-протокол решения задачи*** (запись протокола решения задачи на естественном языке)
  - ***ея-протокол доказательства***

Перечислим некоторые отношения, заданные на множестве информационных конструкций общего вида:
- ***семантическая эквивалентность информационных конструкций***
- ***информационная конструкция – ключевое понятие***
  - ***портрет – объект***
  - ***определение – определяемое понятие***
  - ***пояснение – поясняемое понятие***
- ***последовательность информационных конструкций*** (порядок просмотра информационных конструкций в рамках более крупной информационной конструкции)
- ***разбиение информационных конструкций*** (разбиение на информационные конструкции, являющиеся фрагментами)
- ***информационная конструкция – автор***
- ***информационная конструкция – рецензент***
- ***информационная конструкция – издание***

Перечислим некоторые отношения, заданные на множестве текстов:
- ***семантическое включение*** (когда информация, содержащаяся в одном тексте, содержится также и в другом);
- ***семантическая эквивалентность текстов*** (перевод с одного языка на другой, с одной формы на другую);
- ***текст-ключевое понятие***;
- ***определение – определяемое понятие***;
- ***пояснение-поясняемое понятие***;
- ***комментарий – комментируемое понятие***;
- ***понятие – пример*** (описание какого-либо примера);
- ***включение текста*** (в частности, это может быть связь между некоторым библиографическим источником и взятой из него цитатой);
- ***последовательность текстов*** (не обязательно линейная);
- ***разбиение текста*** (книги – на разделы, разделы (главы) – на подразделы, подразделы – на пункты);

- ***синтаксическая эквивалентность текстов*** .

Перечислим некоторые отношения, заданные на множестве документов:
- ***документ – автор***
- ***документ – издательство***
- ***документ – рецензент***

Отношение **"*синтаксическая эквивалентность текстов*"** – бинарное неориентированное отношение, каждая связка которого связывает два scb-узла, содержимым которых являются информационные конструкции, имеющие абсолютно одинаковый вид. Эти конструкции могут принадлежать одному и тому же языку или разным языкам. Особое значение имеют синтаксически одинаковые конструкции, имеющие разный смысл.

Каждая связка бинарного неориентированного отношения **"*семантическая эквивалентность текстов*"** связывает либо scb-узел, обозначающий атомарное или неатомарное высказывание , представленное на языке SCL, с scb-узлом, содержимым которого является записанный на каком-либо языке текст, семантически эквивалентный указанному выше высказыванию, либо два scb-узла, содержимым которых являются семантически эквивалентные тексты, записанные на одном и том же языке или на разных языках.

Кроме содержимого sc-узлов важнейшей составляющей текстов языка SC также являются идентификаторы sc-элементов, представляющие собой обычные строки символов. В ходе решения задач в sc-машине эти идентификаторы никак не используются. Они необходимы для организации ввода / вывода информации и для организации интеграции различных баз знаний. Для выполнения этих действий необходимо учитывать то, что называется **синонимией** и **омонимией**.

Определение 6.4.1. Два разных sc-элемента считаются **синонимичными** (семантически эквивалентными) в том и только в том случае, если они представляют собой либо один и тот же знак, либо одну и ту же переменную, либо знак одного и того же множества.

Из двух синонимичных sc-элементов либо один, либо оба могут не иметь идентификаторов. Если оба синонимичных sc-элемента имеют идентификаторы, то возможны два случая:
- **тривиальная синонимия**, когда идентификаторы синонимичных sc-элементов совпадают,
- **нетривиальная синонимия**, когда синонимичные sc-элементы имеют разные идентификаторы.

Следует четко отличать синонимию sc-элементов от семантической эквивалентности различных текстов.

В языке SC синонимичные sc-элементы связываются парами бинарного отношения ***"пара синонимии"*** (см. раздел 2). В случае тривиальной синонимии пары синонимии проводятся по умолчанию. То есть все sc-элементы с одинаковыми идентификаторами по умолчанию считаются синонимичными.

Бинарное отношение ***"пара синонимии"*** является отношением эквивалентности. Для любого отношения эквивалентности можно построить отношение, осуществляющее разбиение области определения отношения эквивалентности на классы эквивалентности. Сделаем это для отношения ***"пара синонимии"*** и построим отношение **"*синонимичные sc-элементы*"** (= ***синонимы***). Каждая связка этого отношения связывает все sc-элементы, являющиеся синонимичными некоторому условно выделенному sc-элементу.

Примечание 1. В область определения отношения **"*пара синонимии*"** и отношения **"*синонимичные sc-элементы*"** входят не только константы (знаки множеств), но и переменные, т.е. синонимичными могут быть не только sc-константы, но и sc-переменные.

Примечание 2. Из синонимии двух знаков множеств следует равенство этих множеств. Но обратное, вообще говоря, не верно. То есть могут существовать равные (то есть состоящие из одних и тех же элементов), но разные множества. И, следовательно, знаки, обозначающие эти разные множества не могут считаться синонимичными.

Примечание 3. В языке SC синонимия sc-элементов нужна:
1) для того, чтобы в логических формулах можно было указывать факт совпадения значения некоторой переменной с некоторой константой или со значением другой переменной. При этом такую переменную пару синонимии следует отличать от константной пары синонимии, связывающей два синонимичных переменных sc-элементов;
2) для того, чтобы рассмотреть множество различных вариантов именования какого-либо объекта (идентификаторы синонимичных sc-элементов и есть синонимичные имена одного и того же объекта).

Введем ключевой узел ***"главный синоним"***, обозначающий множество sc-элементов, каждый из которых является главным по отношению к своим синонимам. Поскольку отношение ***"синонимичные sc-элементы"*** является отношением эквивалентности, указание главного узла (среди синонимов) совсем не обязательно осуществлять с помощью атрибута.

Определение 6.4.2. Два разных sc-элемента считаются **омонимичными** в том и только в том случае, если они представляют собой либо разные знаки, либо разные переменные.

Как и синонимия, омонимия sc-элементов бывает тривиальной и нетривиальной. Если два омонимичных sc-элемента имеют разные идентификаторы, то будем их называть **тривиально омонимичными**. Если два омонимичных sc-элемента имеют одинаковые идентификаторы, то будем их называть **нетривиально омонимичными**.

Примечание. Если для двух sc-элементов либо один из них не имеет идентификатора, либо оба не имеют идентификаторов, либо они имеют разные идентификаторы, то по умолчанию (если не оговорено обратное) считается, что указанные sc-элементы являются омонимичными.

Итак, в языке SC допускается использование синонимичных и омонимичных sc-элементов, но при этом должны выполняться следующие правила.

Правило 6.4.1. Если у какого-либо sc-элемента имеются синонимичные ему sc-элементы, то из всей этой группы синонимов должен быть выделен главный синоним с помощью ключевого sc-узла ***"главный синоним"***.

Правило 6.4.2. Нетривиально омонимичные sc-элементы должны быть явно указаны с помощью бинарного отношения ***"пара омонимии"***. Как и для синонимичных sc-элементов, построим отношение ***"омонимичные sc-элементы"*** (= ***омонимы***), осуществляющее разбиение области определения отношения омонимии на классы омонимии. Каждая связка этого отношения связывает все sc-элементы, являющиеся омонимичными некоторому условно выделенному sc-элементу.

Правило 6.4.3. Если построить множество из sc-элементов, являющихся главными синонимами, и из sc-элементов, которые не имеют синонимов (в текущий момент времени), то в этом множестве не должно оказаться ни синонимичных, ни омонимичных sc-элементов.

Для выполнения последнего правила необходимо по крайней мере для одного из двух омонимичных sc-элементов строить синонимичный ему sc-элемент, объявляемый как главный синоним.

Таким образом, борьба с синонимией и омонимией в языке SC осуществляется не путем их искоренения, а путем четкой фиксации синонимичных и омонимичных sc-элементов (если таковые sc-элементы возникает необходимость вводить) и путем организации корректного и безопасного использования таких sc-элементов.

Наиболее актуальным применением гипертекстовых семантических сетей являются электронные учебники нового поколения, в которых используются не только гипертекстовые и мультимедийные технологии, но и глубокая семантическая структуризация учебного материала. Такие электронные учебники будет называть ассоциативными электронными учебниками. Подробно они рассмотрены в [236] (***ИнтелОСиВУО-2001кн***)

6.5. Принципы представления нейросетевых моделей

Ключевые понятия и идентификаторы ключевых узлов: нейросетевая модель, **нейрон**.

Графодинамический подход к представлению и переработке информации можно использовать для интерпретации и интеграции различных моделей представления и переработки информации, в частности, для интеграции нейросетевых моделей переработки информации. Ниже будет рассмотрен пример реализации нейросетевых моделей особого класса **псевдооптических нейронных сетей** (ПНС).

Следует обратить внимание, что модели ПНС ориентированы на моделирование быстрых интеллектуальных процессов мозга. Хорошо известно, что существует обширное множество интеллектуальных процессов, которые в медленном мозгу протекают гораздо быстрее, чем в быстром

компьютере. К таким процессам относятся как обработка внешних данных (обработка образных представлений, узнавание, установление сходства), так и сложные внутренние интеллектуальные процессы – быстрое схватывание сути дела, выделение существенного, оперирование сложной ситуацией как целостным представлением. Многие из них не имеют адекватных аналогов в современных методах искусственного интеллекта. Анализ этих процессов приводит к мысли, что в их основе лежат принципиально другие механизмы – несимвольные представления данных и методы их обработки, высокая скорость которых обеспечивается их малой глубиной и сверхвысоким уровнем параллелизма, природа которого мало похожа на параллелизм вычислительных систем [277] (***Кузнецов О.П. 1995ст-НеклаПвИИ***).

Одним из перспективных подходов к исследованию таких механизмов является реализация неоднократно высказывавшейся различными специалистами [630, 609, 410, 20, 188, 678] (***Heerden P.J.1968bk-FoundOEK, Gabor D.1969art-AssocHM, Прибрам К.1975кн-ЯзыкиМ, Арбиб М.А.1976кн-МетафМ, Денисюк Ю.Н.1982ст-НекотПиПГ, Sowa J.F.1984bk-ConceSIP***) гипотезы о сходстве многих информационных процессов мозга с процессами обработки изображений в оптической голографии. В работе [281] (***Кузнецов О.П.1992ст-ГологМОИвНС***) впервые предложен класс нейронных сетей (впоследствии названных псевдооптическими нейронными сетями - ПНС), в которых, как было показано, возможны голографические эффекты. ПНС используют новую модель интерферирующего нейрона, который воспринимает непрерывные периодические сигналы и характеризуется не только порогом, но и дополнительным непрерывным параметром - потенциалом. Значение потенциала может изменяться от нуля до порога под действием входных сигналов, которые суммируются по закону интерференции. При достижении потенциалом порога нейрон генерирует собственный периодический сигнал.

Модели ПНС, рассмотренные в [277; 281; 284] (***Кузнецов О.П.1995ст-НеклаПвИИ; Кузнецов О.П.1992ст-ГологМОИвНС ; Кузнецов О.П..2000ст-ПсевдНС***), являются геометрическими: в них существенны их геометрические характеристики - расстояния между нейронными слоями сети и нейронами внутри слоя, а также геометрические свойства поверхностей, на которых расположены слои. Правда, в работе [277] (***Кузнецов О.П.1995ст-НеклаПИИ***) отмечается, что геометрические характеристики можно заменить задержками на синапсах.

6.5.1. Краткое описание полной прямолинейной модели псевдооптической нейронной сети и методов расчета её поведения

Ключевые понятия и идентификаторы ключевых узлов: псевдооптическая нейронная сеть, **полная прямолинейная модель**.

Здесь мы рассмотрим одну из наиболее изученных моделей ПНС - полную прямолинейную модель, описанную в [280] (***Кузнецов О.П..2000ст-ПсевдНС***). Начнем с описания основной единицы ПНС - интерферирующего нейрона. Оно содержится в [277; ,281; 282; 284] (***Кузнецов О.П.1995ст-НеклаПвИИ ; Кузнецов О.П.1992ст-ГологМОИвНС ; Кузнецов О.П.1993ст-МоделГПОИвНС ; Кузнецов О.П.1996ст-ПсевдНСПМ ;***). Интерферирующий нейрон *N* имеет m_N входов, q_N выходов и характеризуется тремя положительными действительными числами: порогом P_N , выходной интенсивностью I_N и потенциалом $U_N(t)$, зависящим от времени и не превышающим порога. Нейрон может находиться в пассивном состоянии, в котором он воспринимает входные сигналы, или в активном состоянии, в котором он генерирует выходной сигнал. Нейроны соединены между собой однонаправленными волокнами, имеющими две характеристики: длину *d* и скорость *v* прохождения сигналов. Сигнал S_i длительности τ_i - это функция $S_i(t)=I_i s_i(t)$, определенная на интервале длины τ_i , где $s_i(t)$ - периодическая функция с частотой ν_i , а I_i - константа, называемая интенсивностью сигнала. Пример функции s_i - функция $s_i(t) = \sin 2\pi\nu t$. В дальнейшем считаем, что конкретный вид функции $s_i(t)$ несущественен, сигнал S_i полностью определяется тройкой параметров (I_i, ν_i, τ_i), причем все сигналы, поступающие на вход одного нейрона, имеют одинаковую частоту ν. Сигнал S_i возникает в точке волокна в момент t_{1i} и оканчивается в момент $t_{0i}=t_{1i}+\tau_i$, если в этой точке функция S_i определена на интервале $[t_{1i}, t_{0i}]$. Распространение сигнала по волокну с параметрами *d* и *v*

означает, что, если в начальной точке волокна сигнал S_i возник в момент t_1, то в конечной точке он возникнет в момент $t_1 + d/v$. Для сигнала S_i, распространяющегося со скоростью v, введем понятие длины волны $\lambda_i = \frac{v}{\nu_i}$. Если на входе N в момент t_{1i} возник сигнал S_i, а на другом входе в момент t_{1j} возник сигнал S_j, то величину

$$\varphi_{ij} = 2\pi\nu(t_{1j} - t_{1i}) \qquad (6.5.1)$$

назовем разностью фаз между S_i и S_j на входе N. Состоянием входов нейрона в момент t называется вектор $\sigma(t) = (I_1(t), \ldots, I_m(t))$, где $I_j(t) = 0$, если на j-м входе нет сигнала в момент t, и $I_j(t) = I_j$, если на нем есть сигнал с интенсивностью I_j. Величину $I(t)$, вычисляемую по формуле

$$I(t) = \sum_{i \le m} \sum_{j \le m} \sqrt{I_i(t) I_j(t)} \cos\varphi_{ij}, \qquad (6.5.2)$$

назовем суммарной входной интенсивностью в момент t.

Нейрон функционирует следующим образом. Пусть в момент t нейрон N пассивен и имеет потенциал $U_N(t)$, состояние входов $(I_1, \ldots, I_m)$ на отрезке $[t, t']$ постоянно, I - суммарная интенсивность. Тогда

1)если $U_N(t) + I\nu(t' - t) < P_N$, то

$$U_N(t') = U_N(t) + I\nu(t' - t); \qquad (6.5.3)$$

2)в противном случае существует момент t^*, $t < t^* < t'$, такой, что $U_N(t) + I\nu(t^* - t) = P_N$; в момент t^* нейрон становится активным, и на каждом из его q_N выходов возникает сигнал $S_N = (\frac{I_N}{q_N}, \nu, \tau_N)$, где

$$\tau_N = \frac{P_N}{I_N \nu} \qquad (6.5.4)$$

(время разряда равно времени заряда от сигнала с теми же параметрами). В момент $t^* + \tau_N$ нейрон снова переходит в пассивное состояние; $U_N(t^* + \tau_N) = 0$.

Формула (6.5.3) предполагает, что на данном отрезке времени существует фиксированное число входных сигналов. Однако на входах нейрона в разные моменты времени существуют разные сигналы; каждый сигнал S_i возникает в некоторый момент t_{1i}, заканчивается в момент t_{0i} и имеет длительность $\tau_i = t_{0i} - t_{1i}$. Для произвольного временного интервала справедливо следующее утверждение (его доказательство содержится в [282] (***Кузнецов О.П.1993ст-МоделГПОИвНС***)).

Теорема интерференции. Если на интервале $[t, t']$ на вход нейрона N поступило m сигналов и потенциал U_N не превысил порога, то

$$U_N(t+t') = U_N(t) + \nu\left(2\sum_{i,j\le m}\sqrt{I_i I_j}\cos\varphi_{ij}\tau_{ij} + \sum_{i=1}^{m} I_i\tau_i\right), \qquad (6.5.5)$$

где τ_i - длительность сигнала на *i*-м входе, τ_{ij} - длительность одновременного существования сигналов на *i*-м и *j*-м входах., а суммирование ведется по всем неупорядоченным парам (*i, j*).

В первой сумме формулы (6.5.5) из двух пар (*i, j*) и (*j, i*) берется только одна, причем от выбора той или иной пары зависит знак разности фаз φ_{ij}. Но поскольку $\cos\alpha = \cos(-\alpha)$, то для вычислений либо пара (*i, j*) всегда выбирается так, что $\varphi_{ij} \ge 0$, т.е. $t_{1j} \ge t_{1i}$, либо используется более компактный вариант формулы (6.5.5):

$$U_N(t+t') = U_N(t) + \nu\left(\sum_{i\le m}\sum_{j\le m}\sqrt{I_i I_j}\cos\varphi_{ij}\tau_{ij}\right), \qquad (6.5.6)$$

где *i* и *j* независимо принимают все значения от 1 до *m* и, следовательно, сумма содержит и пару (*i, j*), и пару(*j, i*). Второй сумме формулы (6.5.5) в (6.5.6) соответствует сумма пар (*i, i*), учитывая, что $\cos\varphi_{ii}$ =1, а $\tau_{ii} = \tau_i$. Нетрудно видеть, что, если на интервале [*t, t'*] сигналы не меняются, то (6.5.6) переходит в (6.5.3).

Следствие 1. В случае, когда все интенсивности одинаковы и равны *I*, формула (6.5.6) приобретает вид

$$U_N(t+t') = U_N(t) + I\nu\left(\sum_{i\le m}\sum_{j\le m}\cos\varphi_{ij}\tau_{ij}\right)$$

Величина τ_{ij} выражается через разность фаз следующим образом (доказательство дано в [9]): если $0 \le \varphi_{ij} \le 2\pi\nu\tau$, то

$$\tau_{ij} = \tau_{ij} - \frac{\varphi_{ij}}{2\pi\nu}, \text{ если } t_{0j} \ge t_{0i}, \qquad (6.5.7)$$

$\tau_{ij} = \tau_j$, если $t_{0j} < t_{0i}$.

Если все длительности сигналов одинаковы и равны τ, из $t_{1j} > t_{1i}$ следует $t_{0j} > t_{0i}$ и второй случай в (6.5.7) невозможен, т.е. $\tau_{ij} = \tau - \frac{\varphi_{ij}}{2\pi\nu}$. Это приводит к следующему утверждению.

Следствие 2. Если все интенсивности входных сигналов равны *I*, а все их длительности равны τ, то в условиях теоремы

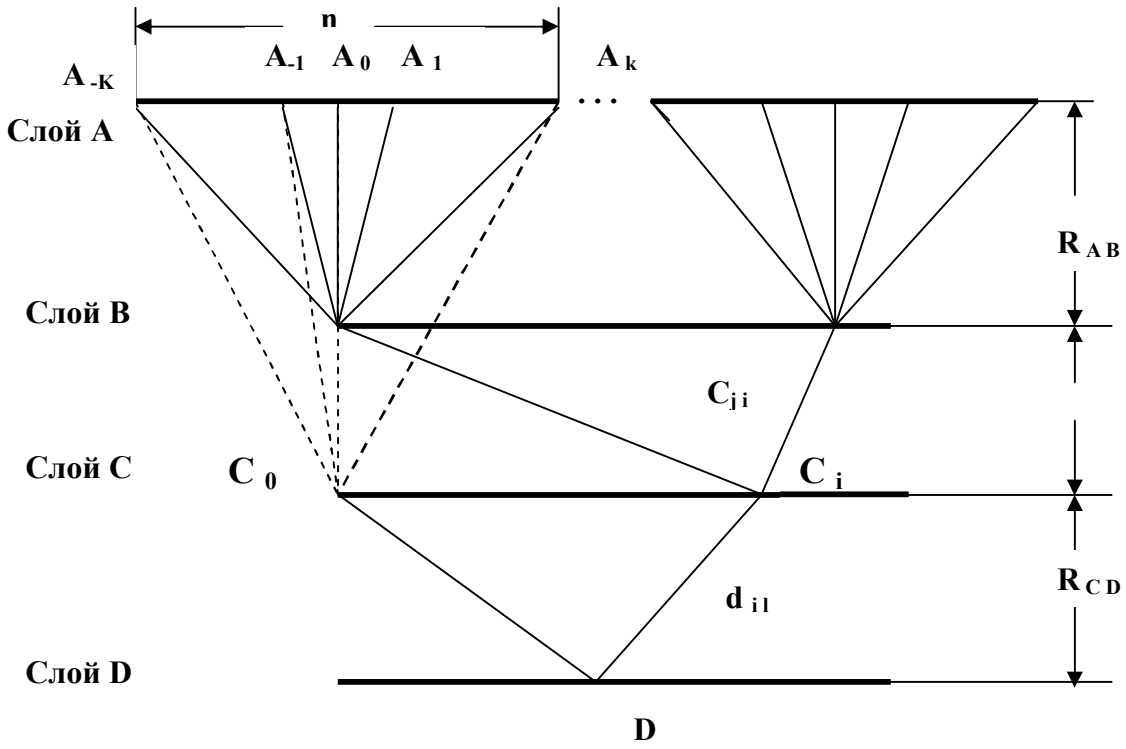
$$U_N(t+t') = U_N(t) + I\nu\left(\sum_{i\le m}\sum_{j\le m}\left(\tau - \frac{\varphi_{ij}}{2\pi\nu}\right)\cos\varphi_{ij}\right), \qquad (6.5.8)$$

где сумма берется по всем *i, j*, таким, что $|\varphi_{ij}| \le 2\pi\nu\tau$.

Поскольку качество голографических эффектов улучшается с увеличением числа нейронов, понижение сложности вычислений при программном моделировании ПНС является важной задачей.

Все геометрические модели ПНС основаны на общей схеме, повторяющей схемы оптической голографии. Эта схема описана в [277; 282] (***Кузнецов О.П.1995ст-НеклаПвИИ ; Кузнецов О.П.1993ст-МоделГПОИвНС***) и содержит четыре нейронных слоя: слой-источник *A*, слой *B*, в котором размещается образ-объект, изображаемый распределением потенциалов его нейронов (чем ближе к порогу потенциал нейрона, тем "ярче" соответствующая точка образа), слой *C* (голограмма), в котором в результате интерференции сигналов от *A* и *B* возникает распределение потенциалов, являющееся голографической записью информации об образе *B*, и слой *D*, в котором после "освещения" голограммы *C* источником *A* восстанавливается образ *B*. Каждый слой – это множество нейронов с одинаковыми параметрами, расположенных на некоторой поверхности на равном расстоянии друг от друга и не связанных между собой.

Рисунок 6.5.1



Из описания этой схемы видно, что выходы *A* должны быть связаны со входами *B,* выходы *A* и *B* – со входами *C*, а выходы *C* – со входами *D*. Скорости и частоты сигналов во всей сети одинаковы. Главная задача при выборе параметров модели – получение в *D* образа, "похожего" на образ в *B*.

Полная прямолинейная модель - это геометрическая модель ПНС со следующими параметрами и свойствами:

1) Все четыре слоя - это прямолинейные параллельные отрезки, лежащие на одной плоскости. Расстояния между ними обозначим через r_{AB} (от *A* до *B*), r_{AC} , r_{BC} , r_{CD} , соответственно, причем $r_{AB} + r_{BC} = r_{AC}$. Следуя принципам оптической голографии (восстановленное изображение объекта находится по другую сторону голограммы на том же расстоянии от нее, на котором находился сам объект), полагаем $r_{BC} = r_{CD}$. Волокна, соединяющие нейроны разных слоев, также прямолинейны. В дальнейшем будем называть их лучами. Все нейроны слоя *A* имеют одинаковые порог P_A и выходную интенсивность I_A . Аналогичные параметры слоев *B,C* обозначаются через P_B , P_C , I_B , I_C соответственно.
2) Число нейронов n_C и n_D в *C* и *D* одинаково: $n_C = n_D = n$, нейроны пронумерованы от 0 до n - 1; расстояния между нейронами одинаковы и равны *e*. Таким образом, отрезок *C* разбит точками C_0, ..., C_{n-1} , в которых находятся нейроны, на n - 1 отрезков длины *e*; длина *C* равна $e(n - 1)$. То же относится и к *D*. Все расстояния измеряются числом волн; или, что то же самое: расстояния измеряются в обычных единицах длины, но λ=1.

3) Отрезок *B* также имеет длину $e(n-1)$ и разбит n точками $B_0, \ldots B_{n-1}$ на $n-1$ отрезков длины e. Однако, в отличие от *C* и *D*, нейроны слоя *B* могут находиться не во всех точках. Номер нейрона слоя *B* - это номер точки, в которой он находится.
4) Коэффициенты ветвления (числа выходов) нейронов слоев *B, C* равны: $q_B = q_C = n$.
5) Параметры слоя *A* выбираются с учетом того, что его излучение должно моделировать два классических случая оптической голографии: точечный источник и плоскую волну. В [284] (***Кузнецов О.П.1996ст-ПсевдНСПМ***) плоская волна моделировалась упрощенным образом: нейрон A_i был соединен только с нейроном C_i (т.е. $q_A = 1$, и интерференция в плоской волне не учитывалась). Это сильно упрощало вычисления, но снижало общность модели. В полной модели это ограничение снимается. Связи *A* с *B* строятся на основе следующих соображений: 1) каждая точка B_i соединена с n точками слоя *A*; совокупность этих n лучей будем называть входным пучком B_i ; 2) геометрия входного пучка B_i не зависит от его номера; этот пучок всегда симметричен относительно перпендикуляра A_iB_i. Это приводит к структуре, показанной на рис.6.5.1. Из нее видно, что в случае плоской волны число нейронов в *A* должно быть больше n, точнее, $n_A \geq 2n$. Аналогичная структура связей имеет место между *A* и *C*. Поэтому для произвольного A_i общее число выходных лучей $q_{Ai} \leq 2n$, но для простоты полагаем всегда $q_{Ai} = q_A = 2n$. Для случая точечного источника $n_A = 1$, причем номер единственного нейрона A_i произволен, но по-прежнему $q_A = 2n$.
6) В начальный момент потенциалы $U_{C\,i} = U_{D\,i} = 0$ для всех нейронов C_i и D_i . В точке B_i нейрон может либо отсутствовать, либо присутствовать. В последнем случае его потенциал U_{Bi} произволен. Распределение потенциалов в слое *B* представляет одномерный образ объекта: нейрон с высоким потенциалом соответствует яркой точке объекта, нейрон с низким потенциалом - темной точке объекта, отсутствие нейрона - отсутствию объекта в данной точке.
7) Другие параметры сети и их обозначения: U_{Ai} , U_{Bj} , U_{Ck} , U_{Dl} - потенциалы нейронов A_i , B_j , C_k , D_l соответственно; S_{Ai} , S_{Bj} , S_{Ck} , S_{Dl} - их выходные сигналы, a_{ij} - расстояние между A_i и B_j (т.е. длина волокна A_i и B_j) , b_{jk} - расстояние между B_j и C_k , c_{ik} - расстояние между A_i и C_k , d_{kl} - расстояние между C_k и D_l .
8) Общая схема сети приведена на рис.6.5.1. Из нее видно, что расстояния a_{ij} , b_{jk} , c_{ik} , d_{kl} вычисляются по теореме Пифагора, например, $b_{0k} = \sqrt{r_{BC}^2 + k^2 e^2}$. Число различных лучей B_jC_k равно n^2, однако число различных расстояний b_{ik} равно n: поскольку $b_{ik} = b_{0,\,|i-k|}$, то массив $\{ b_{00}, \ldots, b_{0,n-1}\}$ содержит все расстояния b_{ik} . Для краткости вместо b_{0k} будем писать b_k . Тогда $b_{ik} = b_{|i-k|}$, в частности, $b_{ii} = b_0 = r_{BC}$. Аналогичные соотношения и обозначения сохраняются и для расстояний между другими парами слоев. Кроме того, $b_{i\,j} = d_{i\,j}$ для всех i, j.

Подробное описание методов расчёта поведения вышеописанной модели псевдооптической нейросети приведено в источнике [280] (***Кузнецов О.П..2000ст-ПсевдНС***).

6.5.2. Принципы представления псевдооптических нейросетей в памяти графодинамических машин

Ключевые понятия и идентификаторы ключевых узлов: **нейрон**, ***слой***, ***нейрон***, ***нейросеть***.

Представление псевдооптических нейронных сетей в графодинамической памяти можно организовать следующим образом. В рассматриваемой нейросетевой модели, как и в большинстве других нейросетевых моделей можно выделить такие понятия, как слой, нейрон, связь. Рассмотрим возможное представление этих понятий в графодинамической памяти.

Каждый нейрон в графодинамической памяти будем представлять связкой:

```
нейрон !; □ процедура_обратного_распространения_ : backward_proc,
            процедура_прямого_распространения_    : forward_proc,
            множество_параметров_                 : par,
            прямые_входные_параметры_             : f_In,
            прямые_выходные_параметры_            : f_Out,
            обратные_входные_параметры_           : b_In,
            обратные_выходные_параметры_          : b_Out □ ;
```

для ПНС множество параметров ***"par"*** содержит порог, потенциал, например:

```
par !; порог_          : 10 ,
       потенциал_ : 8 ,
       количество_выходных_связей_ : 10 ;
```

процедуры ***"forward_proc"*** и ***"backward_proc"*** могут представлять собой программы, реализующие функцию активации и функцию обучения, соответственно, написанные на процедурном языке, который, возможно, является sc-подъязыком, например, на языке SCP.

Каждую связь, как и нейрон, в графодинамической памяти будем представлять связкой:

```
волокно !; □ процедура_обратного_распространения_ : b_fibre_proc,
             процедура_прямого_распространения_   : f_fibre_proc,
             множество_параметров_                : prm,
             прямые_входные_параметры_            : f_IP,
             прямые_выходные_параметры_           : f_OP,
             обратные_входные_параметры_          : b_IP,
             обратные_выходные_параметры_         : b_OP □ ;
```

для ПНС множество параметров ***"prm"*** содержит как неизменные характеристики связи – длину волокна, плотность среды

```
prm !; длина_        : 100 ,
       плотность_ : 1 ;
```

так и характеристики проходящего сигнала: интенсивность, длительность сигнала, пространственно-временное положение:

```
prm !; интенсивность_ : 1 ,
       длительность_  : 10 ,
       время_         : 2 ,
       положение_     : 70 ;
```

"f_IP" и ***"b_IP"*** – прямой выход нейрона-источника и обратный выход нейрона-приёмника соответственно, а ***"f_OP"*** и ***"b_OP"*** – соответственно включаются во множество прямых входных параметров нейрона-приёмника и во множество обратных входных параметров нейрона-источника; процедуры ***"f_fibre_proc"*** и ***"b_fibre_proc"*** строятся аналогичным методом, как и процедуры ***"forward_proc"*** и ***"backward_proc"*** для нейрона. Разумеется, такие процедуры для нейронов источников сигнала , нейронов промежуточных слоёв и нейронов, задающих восстановленный образ, могут быть различны.

Слой нейронной сети задаётся как множество нейронов:

```
слой !; layer1 ;
layer1 !; neuron1 , neuron2 , neuron3 ;
```

т. е. слой ***"layer1"*** состоит из трёх нейронов ***"neuron1", "neuron2", "neuron3"***.

Сеть состоит из слоёв и процедуры обработки поведения сети:

```
нейросеть !; □ 1_ : layer1 ,
              2_ : layer2 ,
              3_ : layer3 ,
              4_ : layer4 ,
              процедура_: net_proc □ ;
```

процедура обработки сети запускается по запросу пользователя и создаёт условие запуска процедур нейронов на первом слое. Те в свою очередь создают условия запуска процедур волокон и нейронов на последующих слоях. Кроме этого, существует набор вспомогательных процедур, которые позволяют автоматическим образом создавать сети определённой конфигурации, менять сохраняемый образ, управлять процессами записи и восстановления образа, сохранять текущее состояние сети. Все эти процедуры объединяются в CASE-систему средств для работы с нейросетями.

Ниже приведём примеры запросов пользователя на построение сетей определённого вида на языке SC:

```
множество связей между слоями_ !;
      [множество связей между слоями_: /"1 "/ ,
      слой, в который идут связи_: /"2 "/],
      [слой, из которого идут связи_: /"1 "/ ,
      слой, в который идут связи_: /"3 "/],
      [слой, из которого идут связи_: /"2 "/ ,
      слой, в который идут связи_: /"3 "/] ;
множество задаваемых начальных потенциалов_ !;
      [для какого слоя задаются потенциалы_: /"2 "/ ,
      [порядковый номер нейрона_: /"1 "/ ,
      начальный потенциал нейрона_: /"2 "/] ,
      [порядковый номер нейрона_: /"2 "/ ,
      начальный потенциал нейрона_: /"0 "/]];
```

Выводы к разделу 6

В разделе 6 были рассмотрены способы представления знаний различных видов. Рассмотрены способы введения различных шкал измерения. Сформулированы принципы описания динамических систем и нейросетевых моделей. Раскрыто понятие гипертекстовой семантической сети, являющейся моделью представления информации в навигационно-поисковой машине (см. раздел 7) и использующейся для систематизации и структуризации знаний о предметной области. Описаны различные виды информационных целей (в виде заданий), которые возможны для графодинамических ассоциативных машин. Отметим, что представление целей в виде заданий позволяет перейти на декларативный формат представления информации в графодинамических машинах, приближая внутренний язык машины к естественному языку. Выстроена иерархия целей, на основе введённых ключевых узлов, когда цель частного вида формируется на основе контекста цели общего вида. Все вышеперечисленное позволяет перейти к рассмотрению различных абстрактных графодинамических ассоциативных машин, в которых будут использованы описанные принципы и способы представления информации, в том числе и служебной (системной) информации, описывающей состояние таких машин.