ИНФОРМАТИКА, ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА И УПРАВЛЕНИЕ

УДК 681.324

С. А. Зинкин

ЭЛЕМЕНТЫ ТЕХНОЛОГИИ ИЕРАРХИЧЕСКОГО КОНЦЕПТУАЛЬНОГО МОДЕЛИРОВАНИЯ И РЕАЛИЗАЦИИ СИСТЕМ И СЕТЕЙ ХРАНЕНИЯ И ОБРАБОТКИ ДАННЫХ

Предлагаются модели и методы, которые могут быть использованы в качестве основы создания новой объектно-ориентированной сетевой технологии моделирования и проектирования распределенных систем хранения и обработки данных на основе согласованных взаимодействий объектов через общее пространство – коммуникационную среду или общее пространство информационных объектов.

Введение

Сетевые представления дискретных динамических систем могут быть положены в основу ряда технологий системного моделирования и распределенного сетевого программирования. Сетевые формализмы также соответствуют парадигме согласования объектов и процессов [1].

В работах [2, 3] были определены сети абстрактных машин (СеАМ) нескольких видов, которые могут быть положены в основу построения новых инструментальных средств для распределенного моделирования и программирования, базирующихся на концепции непосредственно интерпретируемых спецификаций. Последнее означает, что описания моделей «непосредственно» программируются в терминах выражений и операторов некоторого языка или реализуются комплексом специализированных программно-аппаратных средств. При реализации сетевой технологии, базирующейся на формализме СеАМ, требуется размещение в операционной среде вычислительной сети данных, структурированных как объекты некоторого *FS*-пространства (от function spaces).

Концептуально формализм СеАМ берет свое начало от эволюционирующих алгебраических систем [4] и машин Колмогорова [5]. Для построения сетей абстрактных машин в работах [2, 3] была предложена алгебра СеАМ. Использование основных идей из алгебры алгоритмов Глушкова [6], например суперпозиций темпоральных и дополнительных операторов, позволяет естественным образом проектировать сложные иерархические расширенные сети абстрактных машин (ИРСеАМ). В этой связи выделяются два класса сетей абстрактных машин. В относительно простых сетях первого класса подсети, или модули РСеАМ, начиная с тривиальных подсетей – элементарных обновлений интерпретации текущей сигнатуры, составляющих систему образующих, участвуют в операциях однократно. В сложных сетях

второго класса допускается многократное участие указанных подсетей – модулей РСеАМ в произвольных суперпозициях подсетей. В обоих случаях формируются иерархические сети.

В дальнейшем общая аббревиатура СеАМ используется для обозначения технологии или реализации сетей, если это не противоречит контексту; там же, где следует различать конкретные виды выражений, описывающих сети абстрактных машин, используются аббревиатуры СеАМ, РСеАМ и ИРСеАМ.

В приложениях информатики обычно рассматривают некоторое множество Σ представлений с интерпретацией I в множестве S элементов; интерпретация I данному представлению σ ставит в соответствие некоторое абстрактное информационное содержание $I(\sigma)$, т.е. интерпретации соответствует отображение I: $\Sigma \to S$ [7]. Пусть Σ – множество функциональных и предикатных символов различных арностей (в многосортных, или многоосновных, системах тип n-арного предикатного символа – это кортеж $(i_1, i_2, \ldots, i_n, j)$, а тип n-арного предикатного символа – это кортеж $(i_1, i_2, \ldots, i_n)$, где $i_1, i_2, \ldots, i_n, j$ – названия (сорта) для основ, или носителей), S – множество конкретных функций и предикатов. Сеть абстрактных машин СеАМ использует построенные с использованием определенной в работах [2, 3] алгебры модулей абстрактных машин «модули-продукции» и «модули-процедуры», которые модифицируют, или «обновляют», интерпретацию I, выполняя сгруппированные в блоки так называемые правила обновления вида $I(\sigma_i) \leftarrow S_j$. В работах [4, 5] предложено использовать в машинах абстрактных состояний специальные операции – элементарные обновления функций и предикатов. Элементарное правило обновления функции или предиката запишем в виде правила вывода:

$$\frac{t_1, t_2, ..., t_k, t_{k+1}}{s(t_1, t_2, ..., t_k) \leftarrow t_{k+1}},$$

где $t_1, t_2, ..., t_k$ – термы различных сортов; s – функциональный или предикатный символ. В случае, если s – функциональный символ, то t_{k+1} – суть терм любого сорта, а если s – предикатный символ, то t_{k+1} – булево выражение.

Сеть СеАМ функционирует, переходя от одной интерпретации $I(t_i)$ к другой $I(t_k)$, где t_i и t_k – последовательные моменты времени. В алгебре абстрактных машин мы включаем подобные правила обновления в систему образующих.

Представления процедурных и декларативных знаний о предметной области в сценарных моделях

В настоящее время активно развиваются методы, основанные на предварительном формальном описании предметной области – ее понятий, отношений, закономерностей. При формализации предметных областей, связанных с системами и сетями хранения и обработки данных, широко используются сетевые модели. В сетевых моделях представляются как информационно-структурные знания о предметной области, так и знания о процессах, причинно-следственных связях, законах функционирования, сценариях деятельности. Модели представления знаний условно разделяются на декларативные (непроцедурные) и процедурные.

В работах [8, 9] были рассмотрены сценарные модели, являющиеся разновидностью семантических сетей с событиями и темпоральными связями. Самомодифицируемые сценарные модели, реализуемые сетями абстрактных машин высших порядков, особенно удобны при создании такого сетевого программного обеспечения реконфигурируемых систем хранения и обработки структурированных данных, в котором стираются грани между сетевой операционной системой, распределенной системой управления базой данных и распределенным приложением.

Используемые для представления сценариев сети абстрактных машин допускают недвусмысленное описание операционной и функциональной семантик. Кроме того, выражения для модулей, или узлов, СеАМ представляют собой логические формулы, которые аксиоматизируют определенные свойства сети в целом.

Современный подход к переводу модели на язык, приемлемый для используемой ЭВМ, рекомендует поддержку парадигм декларативного и функционального программирования, а также соблюдения основных принципов структурированного, событийно-управляемого и объектно-ориентированного программирования. Технология имитационного моделирования в целом должна поддерживать системную интеграцию и концептуальное проектирование. Концептуальное проектирование обычно выполняется в процессе предпроектных исследований, формулировки технического предложения, разработки эскизного проекта; при концептуальном проектировании применяют ряд спецификаций, среди которых центральное место занимают модели преобразования, хранения и передачи информации [10]. В данной работе рассмотрены четыре класса моделей систем: функциональные модели, описывающие совокупность выполняемых системой функций; информационные модели, отражающие структуры данных, их состав и взаимосвязи; поведенческие модели, описывающие динамику функционирования информационных процессов и оперирующие с такими категориями, как состояния системы, события, переходы из одних состояний в другие, условия перехода, последовательности событий; структурные модели, характеризующие морфологию системы, или ее построение – состав подсистем и их взаимосвязи.

В настоящей работе делается попытка интеграции основных концепций методик, основанных на перечисленных выше четырех классах моделей систем. Выбран формализм сетей абстрактных машин, интерпретация $I(t)$ сигнатуры Σ которых обновляется («эволюционирует» во времени). Помимо обычных отношений, представленных областями истинности соответствующих предикатов, здесь используются темпоральные отношения. Темпоральные отношения и связанные с ними операции могут быть реализованы как следствия причинных воздействий одних активных объектов на другие. Сценарии эволюций интерпретации сигнатуры алгебраической системы задаются согласованной работой сети алгоритмических модулей, или модулей абстрактных машин. Использование логик высших порядков и иерархической организации сетей абстрактных машин расширяет функциональные возможности формализма, который предлагается применять на различных этапах иерархического проектирования и моделирования – от концептуального до разработки исполняемых спецификаций реальной системы.

Структурирование и модификация сценариев

Работа сценария определена над некоторым структурированным *FS*-пространством функций и предикатов. Введем следующие правила вывода для модификации ролевых и объектных предикатов и функций:

$$\frac{t_1, b}{p^{(1)}(t_1) \leftarrow b}, \frac{t_1, t_2, b}{p^{(2)}(t_1, t_2) \leftarrow b}, \frac{t_1, s}{f^{(1)}(t_1) \leftarrow s},$$

где $p^{(1)}$, $p^{(2)}$ – унарный и бинарный предикатные символы; $f^{(1)}$ – унарный функциональный символ; t_1, t_2, s – произвольные термы, возможно, различных сортов; b – булев терм.

Модифицируя предикаты и функции, можно по мере необходимости конкретизировать или изменять отношения между объектами, изменяя их роли и свойства.

Представляет интерес также непосредственное использование одной из основных концепций СеАМ – системы образующих в алгебре модулей, или множества элементарных обновлений функций и предикатов для образования и устранения связей в семантических сетях и сценариях. Например, установлению темпоральной связи «↑» (свидетельствующей о непосредственном следовании подсценариев друг за другом) между подсценариями A_1 и A_2 соответствует элементарное правило обновления бинарного предиката:

$$\uparrow(A_1, A_2) \leftarrow \text{true},$$

а разрыву этой связи соответствует правило обновления:

$$\uparrow(A_1, A_2) \leftarrow \text{false}.$$

Подобные правила определены и над остальными бинарными темпоральными предикатами, каждому из которых соответствует одноименная бинарная темпоральная операция.

Правилу обновления тернарного предиката

$$\text{Op}_\vee(\alpha, A_1, A_2) \leftarrow \text{true}$$

соответствует включение в сценарное выражение α-дизъюнкции в традиционной форме $[\alpha](A_1 \vee A_2)$, а для исключения α-дизъюнкции необходимо выполнить правило обновления

$$\text{Op}_\vee(\alpha, A_1, A_2) \leftarrow \text{false},$$

где $\text{Op}_\vee$ − тернарный предикатный символ, соответствующий префиксной форме записи одноименной операции α-дизъюнкции.

Далее включению подсценария A в цикл и исключению его из цикла соответствуют следующие правила обновления бинарного предиката:

$$\text{Op}_{\{\}}(\alpha, A) \leftarrow \text{true и } \text{Op}_{\{\}}(\alpha, A) \leftarrow \text{false}.$$

Аналогично, здесь $\text{Op}_{\{\}}$ – бинарный предикатный символ, соответствующий префиксной форме записи одноименной операции α-итерации. В сценарные выражения данная операция включается в традиционной форме – $[\alpha]\{A\}$.

Используя подобный подход, можно конструировать произвольные сценарии и определять над ними произвольные процедуры модификации. Формализуя данные понятия, определим над иерархическим сценарием A гипотетическую или виртуальную «суперсеть» абстрактных машин SN (рис. 1).

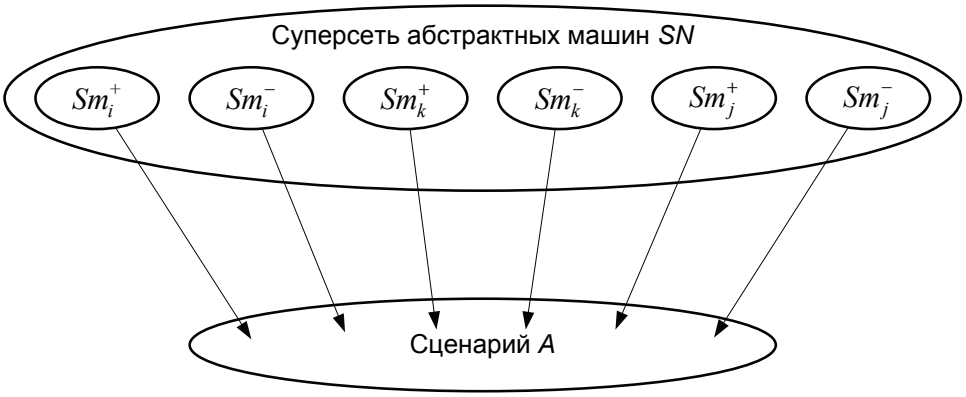

Рис. 1 Формирование и модификация сценария A суперсетью абстрактных машин SN

Данная суперсеть включает в свой состав модули, выполняющие некоторые модификации сценария A:

$Sm_i^+ = [x_i](\mathrm{Op}_\vee(\alpha, A_1, A_2) \leftarrow \mathrm{true} \vee A^E)$;
$Sm_i^- = [z_i](\mathrm{Op}_\vee(\alpha, A_1, A_2) \leftarrow \mathrm{false} \vee A^E)$;
$Sm_k^+ = [x_k](\mathrm{Op}_{\{\}}(\alpha, A) \leftarrow \mathrm{true} \vee A^E)$;
$Sm_k^- = [z_k](\mathrm{Op}_{\{\}}(\alpha, A) \leftarrow \mathrm{false} \vee A^E)$;
$Sm_j^+ = [y_j](\uparrow(A_1, A_2) \leftarrow \mathrm{true} \vee A^E)$;
$Sm_j^- = [y_j](\uparrow(A_1, A_2) \leftarrow \mathrm{true} \vee A^E)$,

где A^E – тождественный сценарий, не выполняющий никаких действий (аналогичный тождественному оператору E в алгебре алгоритмов Глушкова), а в квадратных скобках записаны выражения для условий готовности выбранного фрагмента сценария к обновлению.

Представление сценариев совокупностями предикатов и функций

Развивая объектно-ориентированный подход, представим сценарии совокупностью предикатов (или отношений) и функций (или операций). В используемых нами представлениях сценариев объединяются три основные концепции представления и обработки знаний – логическая, структурная и процедурная. Логическая концепция используется при формировании булевых выражений для α-условий в операциях α-дизъюнкции и α-итерации. Структурной концепции соответствует формирование объектных отношений, характерных для любой семантической сети. Процедурная составляющая в нашем случае базируется на бинарных темпоральных отношениях, включаемых нами в состав фундаментальных отношений сценарной сети наравне с объектными. Кроме бинарных темпоральных отношений вида p_τ и операций вида f_τ, процедурная составляющая включает операции α-дизъюнкции и α-итерации для всюду определенных α-условий, представимые двухсортными тернарным и бинарным ($p_\vee$ и $p_{\{\}}$) предикатами и операциями ($f_\vee$ и $f_{\{\}}$) соответственно (имеются в виду сорта условий и сценариев). Предикаты вида p_τ, $p_\vee$ и

$p_{\{\}}$ задают структуру сценария на всех этапах его построения, а функциями вида f_τ, $f_\vee$ и $f_{\{\}}$ задается собственно операционная семантика сценария. Таким образом, сигнатура, определенная для предметной области, расширяется множеством выражений следующего вида:

p_τ: $S\times S \to$ {true, false};
f_τ: $S\times S \to S$;
$p_\vee$: $L\times S\times S \to$ {true, false};
$f_\vee$: $L\times S\times S \to S$;
$p_{\{\}}$: $L\times S \to$ {true, false},
$f_{\{\}}$: $L\times S \to S$,

где S – множество подсценариев; L – множество логических условий; τ – символ темпоральной операции или отношения.

Представим конкретизации операций и отношений для некоторых наиболее характерных фрагментов какого-либо подсценария. Например, выражение для подсценария L_0 некоторого сценария S_0 запишем в виде

$$L_0 = [\beta_2](S_8 \vee L_1)\uparrow S_{12},$$

где $L_1 = (((S_5\uparrow S_7)\uparrow S_{10})\uparrow S_{11})|^c(((S_5\uparrow S_6)\uparrow S_9)\uparrow S_{11})|^c(S_7||S_9)$.

Здесь для объединения причинно-зависимых подсценариев явно использована операция «$|^c$» конкурентного выполнения. Результатом ее выполнения является теоретико-множественное объединение подграфов, представленных соответствующими подсценариями. Операция «||» используется для синхронизации подсценариев; при этом требуется обязательное пересечение интервалов их реализации. Этапы формирования подсценария L_0 с использованием темпоральных операторов и оператора α-дизъюнкции представлены в табл. 1.

Таблица 1

n	Суперпозиции операторов алгебры сценариев	Конкретизации предикатов
1	$f_\uparrow(S_5, S_7)$	$p_\uparrow(S_5, S_7) \leftarrow$ true
2	$f_\uparrow(f_\uparrow(S_5, S_7), S_{10})$	$p_\uparrow(f_\uparrow(S_5, S_7), S_{10}) \leftarrow$ true
3	$f_\uparrow(f_\uparrow(f_\uparrow(S_5, S_7), S_{10}), S_{11})$	$p_\uparrow(f_\uparrow(f_\uparrow(S_5, S_7), S_{10}), S_{11}) \leftarrow$ true
4	$f_\uparrow(S_5, S_6)$	$p_\uparrow(S_5, S_6) \leftarrow$ true
5	$f_\uparrow(f_\uparrow(S_5, S_6), S_9)$	$p_\uparrow(f_\uparrow(S_5, S_6), S_9) \leftarrow$ true
6	$f_\uparrow(f_\uparrow(f_\uparrow(S_5, S_6), S_9), S_{11})$	$p_\uparrow(f_\uparrow(f_\uparrow(S_5, S_6), S_9), S_{11}) \leftarrow$ true
7	$f_\|(S_7, S_9)$	$p_\|(S_7, S_9) \leftarrow$ true
8	$f_{\|^c}(f_\uparrow(f_\uparrow(f_\uparrow(S_5, S_7), S_{10}), S_{11}), f_\uparrow(f_\uparrow(f_\uparrow(S_5, S_6), S_9), S_{11}))$	$p_{\|^c}(f_\uparrow(f_\uparrow(f_\uparrow(S_5, S_7), S_{10}), S_{11}), f_\uparrow(f_\uparrow(f_\uparrow(S_5, S_6), S_9), S_{11})) \leftarrow$ true
9	$f_{\|^c}(f_{\|^c}(f_\uparrow(f_\uparrow(f_\uparrow(S_5, S_7), S_{10}), S_{11}), f_\uparrow(f_\uparrow(f_\uparrow(S_5, S_6), S_9), S_{11})), f_\|(S_7, S_9)) = L_1$	$p_{\|^c}(f_{\|^c}(f_\uparrow(f_\uparrow(f_\uparrow(S_5, S_7), S_{10}), S_{11}), f_\uparrow(f_\uparrow(f_\uparrow(S_5, S_6), S_9), S_{11})), f_\|(S_7, S_9)) \leftarrow$ true
10	$f_\vee(\beta_2, S_8, f_{\|^c}(f_{\|^c}(f_\uparrow(f_\uparrow(f_\uparrow(S_5, S_7), S_{10}), S_{11}), f_\uparrow(f_\uparrow(f_\uparrow(S_5, S_6), S_9), S_{11})), f_\|(S_7, S_9)))$	$p_\vee(\beta_2, S_8, f_{\|^c}(f_{\|^c}(f_\uparrow(f_\uparrow(f_\uparrow(S_5, S_7), S_{10}), S_{11}), f_\uparrow(f_\uparrow(f_\uparrow(S_5, S_6), S_9), S_{11})), f_\|(S_7, S_9))) \leftarrow$ true
11	$f_\uparrow(f_\vee(\beta_2, S_8, f_{\|^c}(f_{\|^c}(f_\uparrow(f_\uparrow(f_\uparrow(S_5, S_7), S_{10}), S_{11}), f_\uparrow(f_\uparrow(f_\uparrow(S_5, S_6), S_9), S_{11})), f_\|(S_7, S_9))), S_{12}) = L_0$	$p_\uparrow(f_\vee(\beta_2, S_8, f_{\|^c}(f_{\|^c}(f_\uparrow(f_\uparrow(f_\uparrow(S_5, S_7), S_{10}), S_{11}), f_\uparrow(f_\uparrow(f_\uparrow(S_5, S_6), S_9), S_{11})), f_\|(S_7, S_9))), S_{12}) \leftarrow$ true

Вводя дополнительные обозначения для формируемых операторов, упростим выражения в табл. 1 и представим их в табл. 2.

Таблица 2

n	Суперпозиции операторов алгебры сценариев	Конкретизации предикатов
1	$f_{\uparrow}(S_5, S_7) = R_1$	$p_{\uparrow}(S_5, S_7) \leftarrow \text{true}$
2	$f_{\uparrow}(R_1, S_{10}) = R_2$	$p_{\uparrow}(R_1, S_{10}) \leftarrow \text{true}$
3	$f_{\uparrow}(R_2, S_{11}) = R_3$	$p_{\uparrow}(R_2, S_{11}) \leftarrow \text{true}$
4	$f_{\uparrow}(S_5, S_6) = R_4$	$p_{\uparrow}(S_5, S_6) \leftarrow \text{true}$
5	$f_{\uparrow}(R_4, S_9) = R_5$	$p_{\uparrow}(R_4, S_9) \leftarrow \text{true}$
6	$f_{\uparrow}(R_5, S_{11}) = R_6$	$p_{\uparrow}(R_5, S_{11}) \leftarrow \text{true}$
7	$f_{\|}(S_7, S_9) = R_7$	$p_{\|}(S_7, S_9) \leftarrow \text{true}$
8	$f_{\|c}(R_3, R_6) = R_8$	$p_{\|c}(R_3, R_6) \leftarrow \text{true}$
9	$f_{\|c}(R_8, R_7) = R_9$	$p_{\|c}(R_8, R_7) \leftarrow \text{true}$
10	$f_{\vee}(\beta_2, S_8, R_9) = R_{10}$	$p_{\vee}(\beta_2, S_8, R_9) \leftarrow \text{true}$
11	$f_{\uparrow}(R_{10}, S_{12}) = L_0$	$p_{\uparrow}(R_{10}, S_{12}) \leftarrow \text{true}$

Подсценарий L_0 сформирован и готов к реализации при истинности выражения

$$\mu_{L_0} = p_{\uparrow}(S_5, S_7)\&p_{\uparrow}(R_1, S_{10})\&p_{\uparrow}(R_2, S_{11})\&p_{\uparrow}(S_5, S_6)\&p_{\uparrow}(R_4, S_9)\&p_{\uparrow}(R_5, S_{11})\&$$
$$\&p_{\|}(S_7, S_9)\&p_{\|c}(R_3, R_6)\&p_{\|c}(R_8, R_7)\&p_{\vee}(\beta_2, S_8, R_9)\&p_{\uparrow}(R_{10}, S_{12}).$$

Этапы формирования подсценария $L_2 = S_{17}|[\alpha_1]\{(S_{14}\uparrow S_{15})\uparrow S_{16}\}$ некоторого сценария S_0 с операцией α-итерации проиллюстрированы в табл. 3.

Таблица 3

n	Суперпозиции операторов алгебры сценариев	Конкретизации предикатов
1	$f_{\uparrow}(S_{14}, S_{15})$	$p_{\uparrow}(S_{14}, S_{15}) \leftarrow \text{true}$
2	$f_{\uparrow}(f_{\uparrow}(S_{14}, S_{15}), S_{16})$	$p_{\uparrow}(f_{\uparrow}(S_{14}, S_{15}), S_{16}) \leftarrow \text{true}$
3	$f_{\{\}}(\alpha_1, f_{\uparrow}(f_{\uparrow}(S_{14}, S_{15}), S_{16}))$	$p_{\{\}}(\alpha_1, f_{\uparrow}(f_{\uparrow}(S_{14}, S_{15}), S_{16}) \leftarrow \text{true}$
4	$f_{\|}(S_{17}, f_{\{\}}(\alpha_1, f_{\uparrow}(f_{\uparrow}(S_{14}, S_{15}), S_{16}))) = L_2$	$p_{\|}(S_{17}, f_{\{\}}(\alpha_1, f_{\uparrow}(f_{\uparrow}(S_{14}, S_{15}), S_{16}))) \leftarrow \text{true}$

Аналогично предыдущему примеру введем дополнительные обозначения для формируемых операторов, упростим выражения в табл. 3 и представим их в табл. 4.

Таблица 4

n	Суперпозиции операторов алгебры сценариев	Конкретизации предикатов
1	$f_{\uparrow}(S_{14}, S_{15}) = Q_1$	$p_{\uparrow}(S_{14}, S_{15}) \leftarrow \text{true}$
2	$f_{\uparrow}(Q_1, S_{16}) = Q_2$	$p_{\uparrow}(Q_1, S_{16}) \leftarrow \text{true}$
3	$f_{\{\}}(\alpha_1, Q_2) = Q_3$	$p_{\{\}}(\alpha_1, Q_2) \leftarrow \text{true}$
4	$f_{\|}(S_{17}, Q_3) = Q_4 = L_2$	$p_{\|}(S_{17}, Q_3) \leftarrow \text{true}$

Подсценарий L_2 сформирован и готов к реализации при истинности выражения

$$\mu_{L_2} = p_{\uparrow}(S_{14}, S_{15})\&p_{\uparrow}(Q_1, S_{16})\&p_{\{\}}(\alpha_1, Q_2)\&p_{\|}(S_{17}, Q_3).$$

Иерархическое объектно-ориентированное проектирование и сети абстрактных машин первого и высших порядков

Распространяя основные идеи, связанные с использованием механизмов наследования и логического вывода на семантических сетях [11–13], на сценарные модели, включим в состав фундаментальных отношений темпоральные отношения наряду с отношениями вида «класс–подкласс», «класс–суперкласс», «является частью», «подмножество–множество» и др. Известно [11], что возможность «сжатия» базы данных, созданной для представления и хранения семантической сети, обеспечивается, например, транзитивностью родовидового отношения «a_kind_of» (или АКО) и отношения принадлежности части объекта к целому «a_part_of» (или АРО), а операции модификации базы знаний на семантических сетях сводятся к удалению и добавлению новых вершин и ребер. Другим мощным средством манипулирования знаниями являются операции «сопоставления с образцом» [11, 13, 14], когда поиск ответа на запрос реализуется путем сопоставления сети запроса с фрагментами семантической сети. Операции сопоставления с образцом при учете не представленных в сценарных моделях фундаментальных отношений могут быть описаны выражениями СеАМ и РСеАМ для сетей абстрактных машин с использованием логики высших порядков или реализованы путем использования обычных механизмов наследования и логического вывода.

В рамках предлагаемых подходов мы рассматриваем иерархию концептуальных моделей различной степени сложности – от концептуальных моделей, «существующих» в уме специалиста, до сложных имитационных моделей, которые воспроизводят поведение системы в деталях. Построение имитационных моделей на основе сетей абстрактных машин позволяет в широкой степени использовать правила модульного иерархического проектирования; межмодульные связи по управлению и по данным унифицируются и четко формализуются, что позволяет использовать иерархическую имитационную модель непосредственно в реализации проекта, например, в виде сложного комплекса управляющих и функциональных программ в сети хранения и обработки данных. В результате возможно значительное снижение трудоемкости и ускорение процесса проектирования системы в целом, упрощается создание ее компонент. Таким образом, в данной работе предлагается некоторая совокупность конструктивных методов проектирования систем и сетей хранения и обработки данных, рассматриваемых как сложные системы. При анализе и построении сложных систем особенно плодотворными оказались методы объектно-ориентированного проектирования (ООП). Данные методы послужили основой для интеграции различных методов представления знаний. Рассмотрим с позиций объектно-ориентированного подхода некоторые особенности используемых или предлагаемых нами имитационных, сценарных и логико-алгебраических моделей.

Объектно-ориентированный подход к проектированию систем и сетей хранения и обработки данных позволяет реализовать как процедурные методы представления знаний, так и методы, основанные на семантических и сценарных сетях. Кроме того, при использовании объектно-ориентированного подхода можно реализовать и логические методы представления знаний. Формулы, как составные, так и атомарные, могут при этом рассматриваться в

качестве объектов. Некоторые известные недостатки логических методов представления знаний можно преодолеть путем структурирования наборов формул. В качестве другого подхода к преодолению данных недостатков нами предлагается использование логики высших порядков.

В системах искусственного интеллекта традиционно используются декларативные и процедурные методы представления знаний. Как в декларативных, так и в процедурных методах знания представляются структурированными данными. В дальнейшем мы будем уделять большее внимание элементам декларативного подхода к представлению процедурных знаний. Процедурные знания мы будем представлять сценарными сетями, или просто сценариями, включающими события и темпоральные отношения между ними. Темпоральные отношения представляются структурированными данными, которыми можно манипулировать в широких пределах. События, или факты, мы будем описывать формулами в логике первого и высших порядков. При этом операции логического вывода, являющиеся основными примитивами для манипуляции знаниями, одинаково применимы как к декларативным, так и к процедурным знаниям, представленным структурированной совокупностью формул.

При объектно-ориентированном подходе предметная область, или проблемная среда, рассматривается как совокупность объектов и связей между ними. В объектно-ориентированных представлениях подграфы сценарной сети могут представлять сложные факты или события, связанные с поведением объектов в проблемной среде. Роли элементарных фактов при этом играют факты наличия или отсутствия связей определенного типа между объектами. Как разновидность понятия «семантическая сеть», представление которой в базе данных описано в работе [15], в общем случае сценарную сеть мы также представим совокупностью кортежей произвольных отношений (или областями истинности произвольных предикатов). Отношения задаются на множествах (доменах) значений свойств объектов. Каждая такая совокупность, или ассоциативный кортеж, представляет набор характеристик некоторого базового объекта, информационная модель которого задается так называемым комплексом данных [15].

Рассмотрим способы хранения объектов в информационном пространстве (в *FS*-пространстве функций и предикатов), позволяющие сохранить семантическую информацию. Одной из важнейших задач в объектно-ориентированной системе является обеспечение перманентности, или постоянства хранения объектов.

Обычно в реляционной СУБД для представления иерархии классов необходимо решать следующие задачи [16]:

– создать отношения, соответствующие рассматриваемой иерархии классов;

– предусмотреть возможность преобразования объектов в строки и сохранения этих объектов в отношениях;

– предусмотреть возможность чтения строк из отношений и реконфигурирования объектов.

При решении данной задачи обеспечивается преобразование экземпляров класса, т.е. объектов, в кортежи. В работе [16] был исследован ряд способов преобразования классов в отношения: преобразование каждого класса или подкласса в отношение, преобразование каждого подкласса в отношение

и преобразование иерархии в одно отношение. Отмечено, что для данных способов характерна потеря семантической информации. Предложим новый подход к сохранению семантической информации об объектах, базирующийся на элементах логики предикатов второго порядка и на правилах вывода при хранении и извлечении объектов из информационного *FS*-пространства. Как обычно в многосортном исчислении предикатов, рассмотрим множество выражений вида

$$p\colon X_1 \times X_2 \times \ldots \times X_n \to \{\text{true, false}\},$$

где $X_1, X_2, \ldots, X_n$ – сорта, интерпретируемые как множества; p – n-арный предикатный символ.

Предикаты задают структурные связи между понятиями предметной области, например, между объектами или событиями. Рассмотрим далее предикаты $P_{\text{ISA}}(z_1, z_2)$ и $P_{\text{AKO}}(z_3, z_4)$, где $z_1 \in X$ – предметная переменная, пробегающая по всем элементам множества объектов X; $z_2 \in P$ – предикатная переменная, пробегающая по всем предикатам из множества P всех классов; $z_3 \in P_1$ – предикатная переменная, пробегающая по всем предикатам из множества P_1, представляющего подклассы, а $z_4 \in P_2$ – предикатная переменная, пробегающая по всем предикатам из множества P_2, представляющего суперклассы. Множества предикатов P_1 и P_2, возможно, пересекающиеся, т.к. в общем случае некоторые из рассматриваемых классов могут принадлежать как подклассам, так и суперклассам. Здесь P_{ISA} – бинарный предикат, устанавливающий принадлежность объекта классу, а P_{AKO} – бинарный предикат, устанавливающий соответствие «подкласс–суперкласс».

В отличие от работы [16], мы будем представлять классы областями истинности соответствующих предикатов. Рассмотрим пример иерархии наследования для некоторого класса A и его подклассов B, C и D, заданных предикатами $P_A(x_1, x_2, x_3, x_4, x_5)$, $P_B(x_1, x_6, x_7)$, $P_C(x_1, x_8, x_9)$, $P_D(x_1, x_{10})$. Здесь используются 10 предметных переменных различных сортов (типов). Переменная x_1 выполняет роль первичного ключа. Областями истинности данных предикатов представлены четыре отношения, хранимые в реляционной базе данных. Для сохранения семантической информации (в данном случае информации об иерархии наследования), добавим к данным предикатам логическое выражение

$$P_{\text{AKO}}(P_B, P_A) \,\&\, P_{\text{AKO}}(P_C, P_A) \,\&\, P_{\text{AKO}}(P_D, P_A).$$

Истинность данного выражения свидетельствует о задании отношения наследования.

Далее добавим к предикатам и логическому выражению три правила вывода для представления каждого класса одним предикатом:

$(\forall x_1, x_2, x_3, x_4, x_5, x_6, x_7)[\, P_B(x_1, x_6, x_7) \,\&\, P_{\text{AKO}}(P_B, P_A) \supset$

$\supset P'_B\,(x_1, x_2, x_3, x_4, x_5, x_6, x_7)]$;

$(\forall x_1, x_2, x_3, x_4, x_5, x_8, x_9)[\, P_C(x_1, x_8, x_9) \,\&\, P_{\text{AKO}}(P_C, P_A) \supset$

$\supset P'_C\,(x_1, x_2, x_3, x_4, x_5, x_8, x_9)]$;

$(\forall x_1, x_2, x_3, x_4, x_5, x_{10})[\, P_D(x_1, x_{10}) \,\&\, P_{\text{AKO}}(P_D, P_A) \supset$

$\supset P'_D\,(x_1, x_2, x_3, x_4, x_5, x_{10})]$.

Здесь запись $(\forall x_1, x_2, \ldots, x_n)$ эквивалентна записи $(\forall x_1)(\forall x_2,), \ldots, (\forall x_n)$. Таким образом, для хранения объектов и информации о наследовании

свойств объектов без потери семантической информации, как проиллюстрировано на примере, достаточно сформировать базу знаний, включающую экстенсионал – области истинности предикатов и интенсионал – правила вывода. Приведенные выше правила реализуются модулями СеАМ второго порядка, описываемыми следующими выражениями:

$m_B = [\tilde{\forall}\ (x_1, x_2, x_3, x_4, x_5, x_6, x_7)(P_B(x_1, x_6, x_7)\ \&\ P_{\text{AKO}}(P_B, P_A))]$
$(P'_B(x_1, x_2, x_3, x_4, x_5, x_6, x_7) \leftarrow \text{true} \vee R^E)$;
$m_C = [\tilde{\forall}\ (x_1, x_2, x_3, x_4, x_5, x_8, x_9)(P_C(x_1, x_8, x_9)\ \&\ P_{\text{AKO}}(P_C, P_A))]$
$(P'_C(x_1, x_2, x_3, x_4, x_5, x_8, x_9) \leftarrow \text{true} \vee R^E)$;
$m_D = [\tilde{\forall}\ (x_1, x_2, x_3, x_4, x_5, x_{10})(P_D(x_1, x_{10})\ \&\ P_{\text{AKO}}(P_D, P_A))]$
$(P'_D(x_1, x_2, x_3, x_4, x_5, x_{10}) \leftarrow \text{true} \vee R^E)$.

В общем случае иерархию наследования можно доопределить, используя обычные для семантических сетей правила вывода:

$$(\forall x \in X)(\forall y \in P_1)(\forall z \in P_2)[P_{\text{ISA}}(x, y)\ \&\ P_{\text{AKO}}(y, z) \supset P_{\text{ISA}}(x, z)];$$

$$(\forall u \in P)(\forall v \in P)(\forall w \in P)[P_{\text{AKO}}(u, v)\ \&\ P_{\text{AKO}}(v, w) \supset P_{\text{AKO}}(u, w)].$$

Здесь все переменные, кроме x, предикатные и пробегают по множествам предикатов, представляющих классы. Предметная переменная x пробегает по элементам множества всех экземпляров классов (объектов). Ряд правил мы использовали ранее для выявления некоторых отношений в сценарных моделях. Последние два правила реализуются следующими модулями СеАМ второго порядка:

$m_{\text{ISA}} = [\tilde{\forall}\ (x \in X, y \in P_1, z \in P_2)(P_{\text{ISA}}(x, y)\ \&\ P_{\text{AKO}}(y, z))]$
$(P_{\text{ISA}}(x, z) \leftarrow \text{true} \vee R^E)$;
$m_{\text{AKO}} = [\tilde{\forall}\ (u \in P, v \in P, w \in P)(P_{\text{AKO}}(u, v)\ \&\ P_{\text{AKO}}(v, w))]$
$(P_{\text{AKO}}(u, w) \leftarrow \text{true} \vee R^E)$.

В общем случае развитая система объектно-ориентированного проектирования должна поддерживать модификацию определений классов, структуры наследования, спецификации атрибутов и методов. Типичные изменения, которые могут потребоваться, сводятся к следующим [16]:

– изменение определения класса;
– изменение атрибутов;
– изменение методов;
– изменение иерархии наследования;
– определение суперкласса для некоторого класса;
– удаление суперкласса для некоторого класса;
– создание и удаление классов, изменение имен классов.

В рамках предлагаемых нами методов (например, конкретизация иерархии наследования, т.е. конкретизация предиката второго порядка P_{AKO} для рассмотренного ранее примера) осуществляется путем выполнения правил обновления:

$$P_{\text{AKO}}(P_B, P_A) \leftarrow \text{true}, P_{\text{AKO}}(P_C, P_A) \leftarrow \text{true}, P_{\text{AKO}}(P_D, P_A) \leftarrow \text{true},$$

а исключение какого-либо подкласса из иерархии, например подкласса C, осуществляется путем применения правила $P_{\text{AKO}}(P_C, P_A) \leftarrow \text{false}$.

Рассмотрим представление сложных структурированных событий в сценариях на примере следующей интерпретации предметной области, опи-

санной ранее предикатами P_A, P_B, P_C и P_D. Пусть $x_1 \in X_1$ – предметная переменная, пробегающая по объектам сорта «событие», а остальные переменные $x_2, x_3, \ldots, x_{10}$ пробегают по соответствующим множествам $X_2, X_3, \ldots, X_{10}$ объектов, участвующих в реализации событий класса A и подклассов B, C и D. Введем далее бинарные предикаты $R_{1,2}, R_{1,3}, \ldots, R_{1,10}$, характеризующие роль объектов сорта X_i ($i = 2, 3, \ldots, 10$) в событиях сорта X_1. Тогда наряду с предикатами P_A, P_B, P_C и P_D определенность всех ролей некоторых выбранных объектов $x'_i \in X_i$ ($i = 2, 3, \ldots, 10$) при реализации ими соответствующего сложного события x'_i можно охарактеризовать выражениями

$$R_{1,2}(x'_1, x'_2) \,\&\, R_{1,3}(x'_1, x'_3) \,\&\, R_{1,4}(x'_1, x'_4) \,\&\, R_{1,5}(x'_1, x'_5);$$

$$R_{1,6}(x'_1, x'_6) \,\&\, R_{1,7}(x'_1, x'_7);$$

$$R_{1,8}(x'_1, x'_8) \,\&\, R_{1,9}(x'_1, x'_9);$$

$$R_{1,10}(x'_1, x'_{10}),$$

или выражениями с предикатными символами второго порядка

$$R_A(R_{1,2}(x'_1, x'_2), R_{1,3}(x'_1, x'_3), R_{1,4}(x'_1, x'_4), R_{1,5}(x'_1, x'_5));$$

$$R_B(R_{1,6}(x'_1, x'_6), R_{1,7}(x'_1, x'_7));$$

$$R_C(R_{1,8}(x'_1, x'_8), R_{1,9}(x'_1, x'_9));$$

$$R_D(R_{1,10}(x'_1, x'_{10})).$$

Для сохранения семантической информации о наследовании свойств в иерархии событий введем конъюнкцию:

$$P'_{\text{AKO}}(R_B, R_A) \,\&\, P'_{\text{AKO}}(R_C, R_A) \,\&\, P'_{\text{AKO}}(R_D, R_A),$$

где P'_{AKO} – предикатный символ третьего порядка.

Для конкретизации предикатов предлагается использовать модули СеАМ первого, второго и третьего порядков, в которых обновляются предикаты соответствующего порядка.

Заключение

На основе введенного определения самомодифицируемого сценария как активной семантической сети, представленной суперпозициями концептуальных графов, предложен алгебраический подход к конструированию сложных иерархических эволюционирующих сценарных сетей, облегчающий создание новых технологий проектирования систем хранения и обработки данных. Принятая парадигма взаимодействия процессов отличается согласованным использованием информационных объектов, представленных сигнатурой многоосновной алгебраической системы, причем структурные и логические связи между понятиями предметной области представляются сигнатурой и формулами в этой сигнатуре. Сигнатура представляет декларативные знания, сценарии – процедурные знания о предметной области. Предложена реализация сценариев сетями абстрактных машин, состоящих из модулей, объединенных причинно-следственными связями, что позволяет эффективно оперировать с моделями вертикально и горизонтально структурированных распределенных систем. Использование логики предикатов первого и высших порядков в существенной степени повышает операционные возможности сценарных моделей, позволяя реализовывать сложные запросы к сети сценариев. В данном представлении сценарии также являются объектами, представляемыми совокупностями кортежей, или ассоциативны-

ми кортежами, хранимыми в основном *FS*-пространстве или в управляющем C*FS*-пространстве (Control Function Space). Подобное представление сценариев допускает их простую и эффективную аппаратную или программную реализацию в процессорах баз данных и на базе модулей ассоциативной памяти.

Список литературы

1. **Таненбаум, Э.** Распределенные системы. Принципы и парадигмы / Э. Таненбаум, М. ван Стен. – СПб. : Питер, 2003. – 878 с.
2. **Зинкин, С. А.** Сети абстрактных машин высших порядков в проектировании систем и сетей хранения и обработки данных (базовый формализм и его расширения) / С. А. Зинкин // Известия высших учебных заведений. Поволжский регион. Технические науки. – 2007. – № 3. – С. 13–22.
3. **Зинкин, С. А.** Сети абстрактных машин высших порядков в проектировании систем и сетей хранения и обработки данных (механизмы интерпретации и варианты использования) / С. А. Зинкин // Известия высших учебных заведений. Поволжский регион. Технические науки. – 2007. – № 4. – С. 37–50.
4. **Gurevich, Y.** Evolving Algebras 1993: Lipari Guide / Y. Gurevich // Specification and Validation Methods / ed. E. Börger. – Oxford University Press, 1995. – P. 9–36.
5. **Gurevich, Y.** On Kolmogorov machines and related issues. The logic in computers science column / Y. Gurevich // Bulletin of Europian Assoc. for Theor. Comp. Science. – 1998. – № 35. – P. 71–82.
6. **Глушков, В. М.** Алгебра. Языки. Программирование / В. М. Глушков, Г. Е. Цейтлин, Е. Л. Ющенко. – Киев : Наукова думка, 1978. – 320 с.
7. **Брой, М.** Информатика. Основополагающее введение / М. Брой. – М. : Диалог-МИФИ (Springer-Lehrbuch), 1996. – Ч. I. – 300 с.
8. **Зинкин, С. А.** Самомодифицируемые сценарные модели функционирования систем и сетей хранения и обработки данных (базовый формализм и темпоральные операции) / С. А. Зинкин // Известия высших учебных заведений. Поволжский регион. Технические науки. – 2007. – № 1. – С. 3–12.
9. **Зинкин, С. А.** Самомодифицируемые сценарные модели функционирования систем и сетей хранения и обработки данных (реализация и свойства сценарных моделей) / С. А. Зинкин // Известия высших учебных заведений. Поволжский регион. Технические науки. – 2007. – № 2. – С. 13–21.
10. **Норенков, И. П.** Подходы к проектированию автоматизированных систем / И. П. Норенков // Наука в образовании : электронное научное издание. – 2005. – №6. – Режим доступа: http://technomag.edu.ru:8001/db/msg/26310.html
11. Перспективы развития вычислительной техники : в 11 кн. / под ред. Ю. М. Смирнова. Кн. 2. Интеллектуализация ЭВМ / Е. С. Кузин, А. И. Ройтман, И. Б. Фоминых, Г. К. Хахалин. – М. : Высшая школа, 1989. – 160 с.
12. Представление и использование знаний / под ред. Х. Уэно, М. Исидзука. – М. : Мир, 1989. – 220 с.
13. **Поспелов, Г. С.** Искусственный интеллект – основы новой информационной технологии / Г. С. Поспелов. – М. : Наука, 1988. – 280 с.
14. Робототехника и гибкие автоматизированные производства : в 9 кн. Кн. 6. Техническая имитация интеллекта / В. М. Назаретов, Д. П. Ким ; под ред. И. М. Макарова. – М. : Высшая школа, 1986. – 144 с.
15. **Калиниченко, Л. А.** Методы и средства интеграции неоднородных баз данных / Л. А. Калиниченко. – М. : Наука, 1983. – 424 с.
16. **Конолли, Т.** Базы данных. Проектирование, реализация и сопровождение. Теория и практика / Т. Конолли, К. Бегг. – М. : Вильямс, 2003. – 1440 с.