Модель и алгоритмы построения учебных курсов

В. Е. Кривцов, Е. А. Ларшин

Целью работы является формализация модели учебного курса и описание алгоритмов построения учебных курсов. Работа основывается на спецификации учебного курса, принятой в стандарте AICC Script версии 3.4 [1].

Данный стандарт широко применяется в электронных обучающих системах (см., например, [2]) для описания навигации между уроками. Для этого, в соответствии с AICC Script, для каждого урока определяется дополнительное условие (пререквизит), содержащий в себе информацию о необходимости предварительного прохождения каких-либо других уроков.

Модель учебного курса

Введем несколько определений.

Урок — минимальная неделимая порция учебной информации, содержащая как сами учебные материалы, так и условия вхождения в них — пререквизит.

Контент — совокупность уроков, для изучения которых не требуется привлечение уроков, не входящих в это множество. Для удобства обозначений и ссылок каждому уроку в контенте может быть присвоено уникальное (в рамках контента) натуральное число n, которое будет служить его идентификатором. Например, $n = 1$. Будем обозначать контент через W, тогда справедливо обозначение $n \in W$.

Статус урока — булева переменная, отражающая степень завершенности урока. Это вычисляемая характеристика. Начальное значение статуса для всех уроков полагается равным 0, однако потом статус может изменяться в зависимости от значений, принимаемых пререквизитом урока. Статус будем обозначать $s(n)$, где n — идентификатор урока.

Пререквизит урока — логическое выражение [1], значение которого определяется статусами других уроков. Пререквизит будем обозначать через $P(n)$, где n — идентификатор урока. Например, $P(6) = s(1)|s(2)$.

[1] В данной работе рассматривается случай, когда в пререквизитах могут присутствовать логические операторы только двух типов: «и» (“and”, “&”) и «или» (“or”, “|”).

Если значение пререквизита равно 1, то в урок можно войти. Отсутствие пререквизита будем обозначать $P(1) = 1$. Если пререквизит отсутствует, то урок всегда доступен для прохождения.

Значения пререквизитов используются при вычислении статусов уроков, а именно: $s(n) = P(n)$ для всех $n \in W$.

С помощью пререквизитов можно уточнить понятие контента, как совокупности уроков, пререквизиты которых не содержат ссылок на уроки, не входящие в контент.

Последовательность уроков — упорядоченное непустое конечное множество уроков без повторений. Последовательность будем обозначать прописными буквами, например: $N = (n_1, n_2, \ldots, n_k)$, где $\forall\, i = \overline{1, \ldots, k}$ элемент последовательности $n_i \in W$.

Курс — последовательность уроков, статусы всех уроков которой станут равными 1 («пройден»), если сначала для любого не имеющего пререквизита урока из этой последовательности положить его статус равным 1, а затем последовательно (в соответствии с заданным порядком следования элементов в последовательности) вычислить статусы остальных уроков на основании их пререквизитов. При этом предполагается, что до начала указанных вычислений статусы уроков в контенте не менялись, т. е. все были равны нулю.

Курс, ведущий в заданный урок, — курс, заканчивающийся заданным уроком. Этот урок будем называть также финальным уроком курса.

Минимальный курс для заданного финального урока — курс, заканчивающийся в заданном уроке и обладающий следующим свойством: при удалении из него любых уроков, за исключением финального, он перестает быть курсом, заканчивающимся в заданном уроке. Будем обозначать множество всех минимальных курсов, отвечающих заданному уроку, через $M(n)$, где n — идентификатор урока.

Цикл — последовательность уроков, в пререквизитах каждого из которых, за исключением первого, есть ссылка на предыдущий урок, а в пререквизите первого — на последний.

Ациклический контент — контент, в котором нет циклов.

Пример 1.

Пусть задан контент

$$W = \{i\}_1^{14} = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14\}.$$

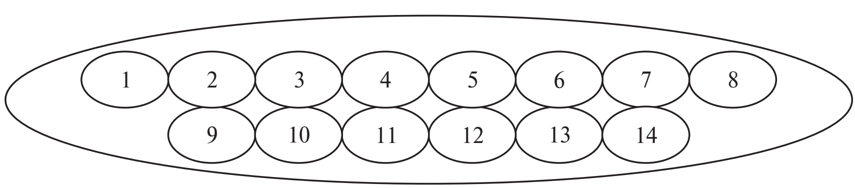


Рис. 1. Пример контента

Пусть уроки имеют следующие пререквизиты:

Урок	Пререквизит	Урок	Пререквизит
1	1	8	$s(6)\&s(7)$
2	1	9	$s(3)\&s(7)$
3	1	10	$s(5)$
4	1	11	$s(7)$
5	1	12	$s(9)\&s(10)\|s(11)$
6	$s(1)\|s(2)$	13	$s(12)$
7	$s(3)\|s(4)$	14	$s(8)\&s(13)$

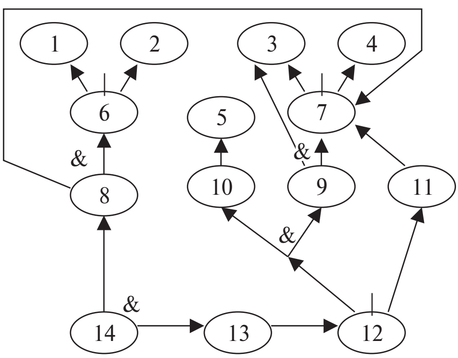


Рис. 2. Отображение пререквизитов в виде И — ИЛИ графа

Эту информацию можно отобразить графически (см. рис. 2). Пусть дана последовательность уроков:

$$N = (2, 4, 6, 7, 11, 12, 13, 14).$$

Она является курсом. Проверим это по определению курса:

1. Статусы всех уроков в контенте равны 0.
2. Для всех уроков из последовательности, не имеющих пререквизитов, положить статус равным 1. В примере это уроки $n_1 = 2$ и $n_2 = 4$.
3. Текущим является урок $n_3 = 6$. Вычисляем его статус: $s(n_3) = s(6) = P(6) = s(1)|s(2) = 0|1 = 1$.
4. Текущим является урок $n_4 = 7$. Вычисляем его статус: $s(n_4) = s(7) = P(7) = s(3)|s(4) = 0|1 = 1$.
5. Текущим является урок $n_5 = 11$. Вычисляем его статус: $s(n_5) = s(11) = P(11) = s(7) = 1$.
6. Теперь для текущего урока $n_6 = 12$ вычисляем его статус: $s(12) = P(12) = s(9)\&s(10)|s(11) = 0\&0|1 = 1$.
7. Вычислим статус урока $n_7 = 13$. Он равен 1, так как $s(13) = P(13) = s(12) = 1$.
8. Вычислим статус последнего урока $n_8 = 14$. Он равен 1, так как $s(14) = P(14) = s(8)\&s(13) = 1\&1 = 1$.

Таким образом, статусы всех уроков последовательности стали равны 1, т. е. последовательность $N = (2, 4, 6, 7, 11, 12, 13, 14)$ является курсом.

Теперь вычислим статусы для последовательности $N = (2, 4, 6, 8, 14)$ из того же контента:

1. Совпадает с аналогичным шагом в предыдущем примере.
2. Для всех уроков без пререквизитов задать статус равным 1. В примере это урок 2 и урок 4.
3. Текущим уроком является урок $n_3 = 6$. Вычисляем его статус: $s(n_3) = s(6) = P(6) = s(1)|s(2) = 0|1 = 1$.
4. Текущим уроком является урок $n_4 = 8$. Вычисляем его статус: $s(8) = P(8) = s(6)\&s(7) = 1\&0 = 0$.

Урок 8 недостижим, так как его пререквизит остался нулевым.

Следовательно, данная последовательность не является курсом, так как не удовлетворяет определению курса. Проверять статусы остальных уроков не имеет смысла.

Алгоритмы построения учебных курсов

В практических приложениях достаточный интерес представляют задачи манипулирования с контентом в целях построения учебных курсов с заданными свойствами. Такими курсами, например, могут быть курсы, обеспечивающие изучение какой-то определенной темы. Особый интерес представляют минимальные курсы, отвечающие заданному финальному уроку. Далее описываются три алгоритма, используемые при решении подобных задач[2)].

Алгоритм 1 решает задачу построения курса, содержащего все уроки заданного контента W. Это достигается путем разбиения контента на последовательные уровни, причем внутри каждого уровня уроки могут изучаться в произвольном порядке.

Уровни определяются рекурсивно:

- уровень 0 состоит из всех уроков, не имеющих пререквизитов;
- уровень $k > 0$ состоит из всех уроков таких, что для каждого из них существует курс, в который входят только уроки с предыдущих уровней, причем в каждом таком курсе обязательно присутствует хотя бы один урок уровня $k - 1$.

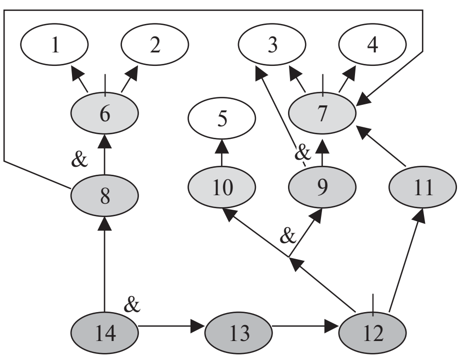
Описание алгоритма 1

1. Статусам всех уроков из W присвоить значение, равное 0.
2. Обозначить через C пустую последовательность.
3. Номер уровня положить равным 0.

[2)] Доказательства сходимости алгоритмов в данной работе не приводятся.

4. Вычислить значения пререквизитов всех уроков из W.
5. Все уроки из W, пререквизиты которых равны 1, отнести к текущему уровню L. Положить их статус равным 1.
6. Упорядочить произвольно уроки, входящие в L. Полученную последовательность приписать к последовательности C, сохранив за построенной последовательностью это же обозначение.
7. Увеличить номер уровня на 1.
8. Определить множество уроков $W = W \backslash L$.
9. Если $W \neq \varnothing$, то перейти к шагу 4, иначе — перейти к шагу 10.
10. Завершить выполнение алгоритма.

Пример 2. Проиллюстрируем работу алгоритма на примере контента из примера 1. Результирующее разбиение представлено в таблице на рис. 3.



Уроки	Уровень
1, 2, 3, 4, 5	0
6, 7, 10	1
8, 9, 11	2
12	3
13	4
14	5

$C = (1, 2, 3, 4, 5, 6, 7, 10, 8, 9, 11, 12, 13, 14).$

Рис. 3. Разбиение контента на уровни

Утверждение о сходимости алгоритма 1. *Если контент ациклический, то алгоритм* 1 *за конечное число шагов построит курс, содержащий все уроки контента.*

Алгоритм 2 предназначен для отбора «нужных» уроков, т. е. уроков, через которые можно попасть в заданный финальный урок.

Введем следующие обозначения:

$L(n)$ — множество уроков, входящих в пререквизит урока n. Если урок n не имеет пререквизита, то $L(n) = \varnothing$. Пример: если $P(12) = s(9)\&s(10)|s(11)$, то $L(12) = \{9, 10, 11\}$.

$L(M)$ — множество уроков (без повторений), входящих в пререквизиты уроков из заданного множества $M = \{n_1, n_2, \ldots, n_k\}$, т. е. $L(M) = \bigcup\limits_{j=1}^{k} L(n_j)$. Пример: $L(\{9, 11\}) = \{3, 7\} \cup \{7\} = \{3, 7\}$.

Описание алгоритма 2

1. Пусть G — искомое множество уроков, положим $G = \varnothing$. Обозначим через t заданный урок.
2. Добавить заданный урок к G.
3. Определим $L = L(t)$.
4. Определим $\underline{L} = L \backslash G$.
5. Добавить все уроки из $\underline{L}$ в G.
6. Построить множество $L = L(\underline{L})$.
7. Пока $L \neq \varnothing$, перейти к шагу 4.
8. Выйти из алгоритма.

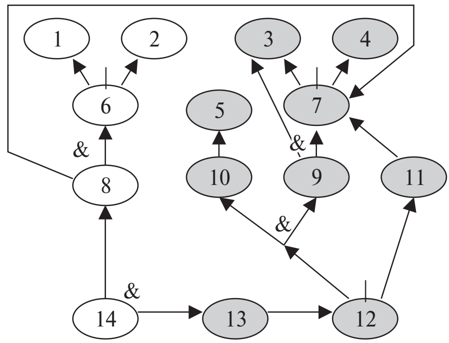


Рис. 4. Поиск необходимых уроков

Пример 3. Проиллюстрируем работу алгоритма на контенте из примера 1, выбрав в качестве финального урок 13.

На рис. 4 заливкой отмечены уроки, отобранные в результате работы алгоритма.

Утверждение о сходимости алгоритма 2. *Если контент ациклический, то алгоритм 2 за конечное число шагов построит множество уроков, каждый их которых входит хотя бы в один курс, ведущий в заданный урок. Других таких уроков в контенте нет.*

Алгоритм 3 построения множества минимальных курсов по заданному финальному уроку.

Введем следующее обозначение:

$$R(n) = \left[P_n(R(n_1), R(n_2), \ldots, R(n_k))\right] \& s(n),$$

где $P_n(s(n_1), s(n_2), \ldots, s(n_k)) = P(n)$, $n_i \in L(n)$. В частности, $R(n) = s(n)$, если $L(n) = \varnothing$.

Описание алгоритма 3

1. Пусть t — заданный финальный урок.
2. Записать $R(t) = \left[P_t(R(n_1), R(n_2), \ldots, R(n_k))\right] \& s(t)$, где $n_i \in L(t)$.
3. Пока в $R(t)$ есть выражения, содержащие $R(n)$, делать подстановку $R(n) = \left[P_n(R(n_1), R(n_2), \ldots, R(n_k))\right] \& s(n)$.
4. Привести $R(t)$ к ДНФ (дизъюнктивной нормальной форме), сохраняя порядок переменных.
5. Рассмотреть $R(t)$ как символьную запись, обозначить ее R.
6. Заменить в R: «$s(z)$» $\rightarrow$ «z», где z — натуральное число; «&» $\rightarrow$ «,»; «|» $\rightarrow$ «;».

7. Сформировать множество минимальных курсов $M(t)$. Его элементами являются последовательности уроков, определяемые участками R, на которые ее делят символы «;».
8. Завершить выполнение алгоритма.

Пример 4. Проиллюстрируем работу алгоритма на контенте из примера 1, выбрав в качестве финального урок 13.

1. Выпишем несколько примеров функции $R(t)$:

$$R(3)=s(3), \quad R(4)=s(4), \quad R(5)=s(5),$$
$$R(7)=P_7(R(3),R(4))\&s(7)=(s(3)|s(4))\&s(7),$$
$$R(10)=P_{10}(R(5))\&s(7)=s(5)\&s(7)$$
$$\dots$$
$$R(13)=P_{13}(R(12))\&s(13)=\dots=[s(5)\&s(10)\&s(3)\&(s(3)|s(4))\&s(7)\& s(9)|(s(3)|s(4))\&s(7)\&s(11)]\&s(12)\&s(13).$$

2. После приведения $R(13)$ к ДНФ, получим:
$R(13) = s(5)\&s(10)\&s(3)\&s(7)\&s(9)\&s(12)\&s(13)|s(5)\&s(10)\&s(3)\& s(4)\&s(7)\&\&s(9)\&s(12)\&s(13)|s(3)\&s(7)\&s(9)\&s(11)\&s(12)\&s(13)| s(4)\&s(7)\&s(9)\&s(11)\&s(12)\&s(13)$.
3. $R = s(5)\&s(10)\&s(3)\&s(7)\&s(9)\&s(12)\&s(13)|s(5)\&s(10)\&s(3)\& s(4)\&s(7)\&\&s(9)\&s(12)\&s(13)|s(3)\&s(7)\&s(9)\&s(11)\&s(12)\&s(13)| s(4)\&s(7)\&s(9)\&s(11)\&s(12)\&s(13)$. После замены получим:
$R=5,10,3,7,9,12,13;5,10,3,4,7,9,12,13;3,7,11,12,13;4,7,11,12,13.$
4. $M(13) = \{\{5, 10, 3, 7, 9, 12, 13\}, \{5, 10, 3, 4, 7, 9, 12, 13\}, \{3, 7, 11, 12, 13\}, \{4, 7, 11, 12, 13\}\}$.

Утверждение о сходимости алгоритма 3. *Если контент ациклический, то алгоритм 3 за конечное число шагов построит множество всех минимальных курсов по заданному финальному уроку.*

Литература

1. AICC/CMI CMI001 Guidelines for Interoperability Version 3.4. October 23, 2000. Includes: AICC Course Structure Format, AICC CMI Data Model (www.aicc.org).
2. SCORM Version 1.2 (www.adlnet.org).