УДК 519.876.3
ББК 22.176 + 65.23

ЭФФЕКТИВНОСТЬ МЕТОДА МОДЕЛИРОВАНИЯ СТРЕЛОЧНОГО СЕТЕВОГО ГРАФИКА ПО МИНИМИЗАЦИИ КОЛИЧЕСТВА ФИКТИВНЫХ РАБОТ

Постовалова И. П.[1]
(*Челябинский филиал Финансового университета при Правительстве РФ, Челябинск*)

На практике встречаются сетевые графики с различной структурой: типа «работы-вершины» и «работы-дуги» (стрелочный сетевой график). Переход от сети типа «работы-дуги» к сопряжённой осуществляется однозначно и без затруднений. Решение обратной задачи неоднозначно, поскольку существуют различные эквивалентные сети типа «работы-дуги», отличающиеся составом событий и фиктивных работ. Сеть типа «работы-дуги» не требует фиктивных операций, если списки опорных операций либо совпадают, либо не пересекаются. В противном случае эти списки проверяются на взаимное вложение с целью уменьшения количества фиктивных операций.

Эффективность метода по уменьшению количества фиктивных работ проверена на нескольких важных классах тестовых задач, охватывающих практически все встречающиеся составные части проектов.

Ключевые слова: сетевая модель, стрелочный сетевой график, график «работы-дуги», фиктивная работа.

Сетевая модель представляет собой план выполнения некоторого комплекса взаимосвязанных работ (операций), заданного

---

[1] *Ирина Павловна Постовалова, кандидат физико-математических наук, доцент* (*ira.postovalova@yandex.ru*).

в специфической форме сети, графическое изображение которой называется сетевым графиком. Сетевой график – это ориентированный граф без контуров (directed acyclic Graph; это английское название иногда сокращают до «dag»), рёбра или вершины которого имеют одну или несколько числовых характеристик. Ориентированные рёбра называются дугами.

Существуют сетевые графики с различной структурой: типа «работы-вершины» (AoN: Activity-on-Node) и «работы-дуги». Последние ещё называются стрелочными сетевыми графиками (AoA: Activity-on-Arrow Network), например, на сайте бизнес - инжиниринговых технологий БИТЕК [6] и в глоссарии проектного менеджера [7] – «это метод построения сетевых моделей, в которых дуги (стрелки) интерпретируются как работы». На стадии разработки удобнее составить сеть AoN, а в процессе управления удобнее пользоваться AoA. Так, например, сеть AoN предпочтительна при частых изменениях состава и структуры проекта, так как отображение этих изменений в AoN производится непосредственно, а в сети AoA может потребовать существенной перестройки. Построение сетей типа AoN предпочтительнее ещё и потому, что не требует введения дополнительных элементов в виде фиктивных работ. Фиктивной работой (зависимостью) называется связь между какими-то результатами работ (событиями), не требующими затрат времени вообще или требующая минимальных затрат времени, не отражаемых в сетевой модели.

Преобразование сети проекта в сопряжённую необходимо также в случае, когда имеющееся математическое обеспечение ориентировано на другой тип сети.

Переход от сети типа AoA к сопряжённой осуществляется однозначно и без затруднений. Решение обратной задачи неоднозначно, поскольку существуют различные эквивалентные стрелочные сетевые графики, отличающиеся составом событий и фиктивных работ, и поэтому требуется структурная оптимизация. Преобразование типа сети легко осуществить растяжением каждой вершины–работы в дугу (j, k), представленную парой номеров начального (j) и конечного (k) событий, принадлежащих множеству вершин новой сети типа AoA. Прежние дуги–

связи называют фиктивными работами [4]. Однако при этом резко увеличивается число узлов и дуг. Фиктивные работы – это просто связи, и функции на них не определены. Количество фиктивных работ стремятся сократить.

В основных положениях по разработке и применению систем СПУ, а также в существующих методах СПУ отсутствуют методы, алгоритмы и программы по построению эффективных сетевых графиков сложных проектов типа «работы-дуги» с минимальным количеством фиктивных работ. Построение сетевых моделей с помощью основных положений базируется на использовании ряда правил, на опыте и знаниях ответственного исполнителя, логически выстраивающего технологические цепочки последовательности работ. При этом имеет место многовариантность и большая трудоемкость процесса проектирования.

Наиболее полные варианты сокращения фиктивных работ предложены Разумовым И. М., Беловой Л. Д., Ипатовым М. И., Проскуряковым А. В. в их совместной работе [5]. Однако их нельзя оценивать, как конечный результат по минимизации фиктивных работ, так как для ряда сетевых графиков возможно меньшее количество фиктивных работ. Другая проблема – обязательно ли первоначально формировать полный список фиктивных операций, а потом его сокращать? Возможно ли создание эффективных алгоритмов с меньшей продолжительностью счёта и меньшим объёмом памяти, а главное, позволяющих вводить малое количество фиктивных работ?

Возникает идея метода преобразования типа в некотором смысле противоположного: вначале добавить только необходимые фиктивные операции, после чего генерировать события.

Среди большого списка просмотренной учебной и научной литературы нечто похожее удалось найти только в работе Гришина А. П. – это «синтез рёберной сетевой модели на основе расширения матрицы бинарных отношений непосредственного предшествования элементарных работ» [2]. Однако приведённый алгоритм излишне сложен и не совсем адекватен. Достаточно сказать, что в этой работе рассмотрен пример орграфа с циклами, который не может быть сетевым графиком, а также сформулирована следующая теорема, требовавшая доработки для сетей проектов: «для того, чтобы рёбра ориентированного графа можно было упорядочить, необходимо и достаточно, чтобы граф был деревом». Графы сетевых проектов не являются деревьями. Тем не менее, их дуги можно упорядочить так, чтобы номер любой дуги, исходящей из любой вершины, был больше номера любой дуги, заходящей в ту же вершину. Богомоловым А. М. [1] доказана исчерпывающая теорема о том, что «в орграфе $\vec{G}$ существует правильная нумерация вершин тогда и только тогда, когда $\vec{G}$ – бесконтурный орграф». В работе Гришина А. П. также необходимо учесть исключение ненужных элементарных фиктивных операций.

Автором статьи под руководством профессора А. Е. Дыхнова был создан новый метод добавления необходимых фиктивных работ на основе исключения пересечений списков предшествующих работ и элементарный метод генерации событий, в котором одинаковым множествам опорных работ соответствует одно событие.

Обычно исходная информация о проекте представляется перечнем операций a_i, $i = 1, \ldots, n$. Для каждой a_i известен список $G(a_i)$ предшествующих операций, чем и определяется сеть типа AoN (см. рис.1)

a_i	$G(a_i)$
a_1	–
a_2	–
a_3	–
a_4	a_1
a_5	a_1, a_2
a_6	a_1, a_2, a_3

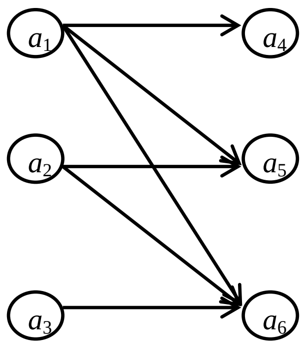
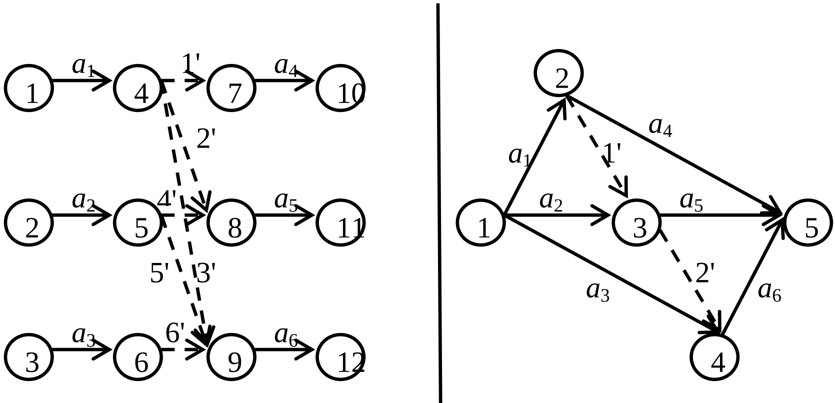


Рис.1. Пример сети AoN

Если после растяжения вершин (см. рис. 2а) применить сокращения из [5], то удастся сократить только две фиктивные операции (1' и 6') и объединить события (1, 2, 3), (10, 11, 12), (4, 7), (6, 9). Результат – 4 фиктивных и 6 событий.

Наш результат (см. рис. 2б) для примера на рис. 1 – всего 2 фиктивных и 5 событий.



6 фиктивных
12 событий
а)

2 фиктивных
5 событий
б)

Рис. 2. Сеть «работы – дуги»:
а) методом растяжения вершин;
б) методом генерации событий

Этот метод предусматривает формирование минимального списка $G_-(a_i)$ непосредственно предшествующих операций, а также полного списка $G_+(a_i)$ всех предшествующих операций.

Заметим, что $G_-(a_i) = \Gamma^{-1}(a_i)$; $G_+(a_i) = Q(a_i) \setminus \{a_i\}$, где $\Gamma(a_i)$ – отображение, совпадающее с минимальным списком последующих работ; $Q(a_i)$ – контрадостижимое множество операции a_i. Построение $G_-(a_i)$ сводится к последовательному просмотру

a_i, $G(a_i)$ и исключению тех $a_j \in G(a_i)$, которые являются дальними предшественниками (G^k) других операций множества $G(a_i)$. С этой целью, пока $G(a_i)$ не стабилизируется, выполняются в цикле следующие действия: $a_j \in G(H_{ij}) \Rightarrow G(a_i) = H_{ij}$, где $H_{ij} = G(a_i) \setminus a_j$.

Сеть типа «работы- дуги» не требует фиктивных операций, если группы опорных операций либо совпадают, либо не пересекаются. В противном случае, списки $G_+(a_i)$ проверяются на взаимное вложение с целью уменьшения количества фиктивных операций. Если $G_+(a_j) \subset G_+(a_k)$, то добавляется всего одна фиктивная операция a', при этом $G\_(a') = G\_(a_j)$, $G\_(a_k)$ заменяется на $a' \cup (G\_(a_k) \setminus G\_(a_j))$.

При отсутствии вложенности добавляются две фиктивные операции a' и a'': $G\_(a') = G\_(a'') = G\_(a_j) \cap G\_(a_k)$; $G\_(a_j)$ и $G\_(a_k)$ заменяются на $a' \cup (G\_(a_j) \setminus G\_(a_k))$ и $a'' \cup (G\_(a_k) \setminus G\_(a_j))$.

После устранения пересечений множеств $G\_(a_i)$ генерируются события по одному на каждую группу $G\_(a_i)$ одинакового состава, начиная с $G\_(a_i) = \varnothing$. Завершающее событие соответствует окончанию операций, для которых $G\_^{-1}(a_i) = \varnothing$.

Пример в таблицах 1 и 2 демонстрирует преимущество предложенной методики проверки взаимного вложения полных списков предшественников и генерации событий исключением пересечений.

Таблица 1. Пример

a_i	$G\_(a_i)$	$G_+(a_i) \setminus G\_(a_i)$
A, B, C	–	–
D	*B*	–
E	*C, D*	*B*
F	*A, D*	*B*
G	*E*	*C, D, B*
H	*A, E*	*C, D, B*
I	*A, G*	*E, C, D, B*
J	*F, C*	*A, D, B*
K	*E, J*	*F, C, A, D, B*
L	*G, K*	*E, J, F, C, A, D, B*

Результаты вычислений приведены в таблице 2, где в графе a_s – полный перечень операций, включая фиктивные; операции с одинаковыми предшественниками сгруппированы так, что начальное событие (j) группы последующих операций совпадает с конечным событием (k) группы предшественников, на что и указывают стрелки в таблице 2. В общем случае работы определяются тройкой чисел (a_s, j, k), где a_s – номер работы, j – номер начального события, k – номер конечного события. Кратные дуги a_s имеют одинаковые инцидентные события j, k, поэтому для них вводится номер дуги s.

При учёте вложенностей $G_+(a_i)$ добавляются 8 фиктивных операций, иначе потребовалось бы 11 фиктивных операций.

Таблица 2. Результаты вычислений

a_s	$G\_(a_s)$	Начальные события (j)	Конечные события (k)
A, B, C	–	0	4, 1, 3
D	B	1	2
E, C'	C, D'	3	5, 8
F, A', A''	A, D''	4	8, 6, 7
G, E', E''	E	5	7, 6, 9
H	A', E'	6	11
I, G'	A'', G	7	11, 10
J	F, C'	8	9
K	E'', J	9	10
L	G', K	10	11
D', D''	D	2	3, 4

На основе метода создана комплексная программа синтеза сетевой модели «работы-дуги», зарегистрированная в отраслевом фонде алгоритмов и программ (ОФАП) [3] и в Информационно-библиотечном фонде РФ. Программа может быть использована на стадии проектирования и в учебном процессе. Проектировщикам не потребуется выявлять и нумеровать события и фиктивные операции, а достаточно только составлять для каждой операции список предшествующих операций. Это уменьшает трудоёмкость и сокращает процесс разработки.

Программа учитывает встречающуюся на практике возможность переопределения отношения порядка, когда пользователь (даже порой искушенный) наряду с необходимыми непосредственно предшествующими операциями указывает по ошибке и некоторые операции дальнего предшествования. Последние выявляются и удаляются.

Эффективность метода по уменьшению количества фиктивных работ проверена на нескольких важных классах тестовых задач, охватывающих практически все встречающиеся составные части проектов: класс задач со Ступенчатым Набором Предшественников из n операций {СНПn}, класс задач с Полным Набором Предшественников из n операций {ПНПn}; класс задач с (n – 1) Элементными Наборами Предшественников из n операций {n – 1ЭНПn}.

В частности, для класса задач со Ступенчатым Набором Предшественников из n операций {СНПn}: $G(a_i)=\varnothing,\quad 1 \le i \le n$;

$$G(a_i) = \bigcup_{j=1}^{i-n} a_j,\ \ n + 1 \le i \le 2$$

, представленного в таблице 3, при преобразовании типа сети добавляется минимальное количество: n – 1 фиктивная работа и n + 2 события вместо $n(n + 1)/2$ фиктивных работ и $4n$ событий при непосредственном преобразовании методом растяжения вершин.

Таблица 3. Данные задач из класса {СНПn}

a_i	$G(a_i)$
a_1	–
a_2	–
⋮	⋮
a_n	–
a_{n+1}	a_1
a_{n+2}	a_1, a_2
⋮	⋮
a_{n+n}	a_1, a_2, …, a_n

Задача из класса {СНПn}, как и любая другая задача, являются частью задачи из класса с Полным Набором Предшественников из n операций {ПНПn}.

Этот класс требует большого количества фиктивных операций. Но и в этом случае предложенный метод приводит к сокращению количества фиктивных работ до минимума. Например, для $n = 3$ данные для проекта представлены в таблице 4, а соответствующий сетевой график «работы-дуги» – в таблице 5.

Таблица 4. Данные проекта {ПНП3}

a_i	$G(a_i)$
1	–
2	–
3	–
4	1
5	2
6	1, 2
7	3
8	1, 3
9	2, 3
10	1, 2, 3

Таблица 5. Оптимальный по структуре стрелочный сетевой график {ПНП3}

a_i	$G(a_i)$	i	j
3, 2, 1	–	0	3, 2, 1
1', 1'', 4	1	1	4, 5, 8
2', 2'', 5	2	2	4, 6, 8
3', 3'', 3''', 6	3	3	5, 6, 7, 8
7', 7	1', 2'	4	7, 8
8	1'', 3'	5	8
9	2'', 3''	6	8
10	7', 3'''	7	8

Для {ПНПn} добавляется всего $2(2^n - n - 1)$ фиктивных операций из возможных $n2^{n-1}$ связей. Их отношение составляет:

(1) $$\frac{2(2^n - n - 1)}{n2^{n-1}} = \frac{4(2^n - n - 1)}{n2^n} = \frac{4}{n}(1 - \frac{n+1}{2^n}) .$$

Результаты по количеству фиктивных работ для {ПНПn} при n = 1, …, 10 отражены в таблице 6, а ниже приводятся пояснения к полученным оценкам для {ПНПn}.

Таблица 6. Результаты предложенного алгоритма для задач класса {ПНПn}

Количество начальных работ	Минимальное количество фиктивных работ	Всего связей
1	0	1
2	2	4
3	8	12
4	22	32
5	52	80
6	114	192
7	240	448
8	494	1024
9	1004	2304
10	2026	5120

Количество подмножеств из n элементов – это $\sum_{i=0}^{n} C_n^i = 2^n$. Фиктивные операции не нужны для начальных операций ($C_n^0 = 1$) и для операций, у которых в предшественниках единственный элемент ($C_n^1 = n$). Итого всего подмножеств с фиктивными операциями: $2^n - n - 1$. Списки предшественников $G(a_i)$ проекта с n операциями (обозначим $G_n(a_i)$) получаем на основе списков $G_{n-1}(a_i)$ проекта с $n - 1$ операциями добавлением n-ой операции отдельно и к каждому $G_{n-1}(a_i)$. В результате такого построения общее число фиктивных работ будет $2(2^n - n - 1)$.

Множество всех связей:

(2) $$\sum_{i=0}^{n} iC_n^i = \sum_{i=0}^{n} \frac{in!}{i!(n-i)!} = \sum_{i=1}^{n} \frac{n(n-1)!}{(i-1)!(n-i)!}! = n\sum_{i=1}^{n} C_{n-1}^{i-1} = n2^{n-1}.$$

Рассмотрим ещё одну задачу с 4-хЭлементными Наборами Предшественников из 5 операций {4ЭНП5}, представленную в таблице 7, где одинаковым спискам $G(a_i)$ в одной строке перечислены операции a_i слева.

Таблица 7. Данные проекта {4ЭНП5}

a_i	$G(a_i)$
5, 4, 3, 2, 1	–
6	1, 2, 3, 4
7	1, 2, 3, 5
8	1, 2, 4, 5
9	1, 3, 4, 5
10	2, 3, 4, 5

Вместо 20 фиктивных достаточно 18 (решение в таблице 8).

Таблица 8. Решение для проекта {4ЭНП5}

a_i	$G(a_i)$
5, 4, 3, 2, 1	–
6	12', 4'
7	12'', 5'
8	11', 13'
9	14', 1'
10	14'', 2'
11'', 11'	1'', 2''
12'', 12'	11'', 3'
13'', 13'	5'', 4''
14'', 14'	13'', 3''
1'', 1'	1
2'', 2'	2
3'', 3'	3
4'', 4'	4
5'', 5'	5

В данной статье впервые докажем также эффективность в уменьшении количества фиктивных работ для класса проектов со всеми опорными работами из различных (n – 1)-подмножеств n начальных работ, сокращенно {n – 1ЭНПn} (n – 1- Элементные Наборы предшественников из n операций).

Анализ решений задач из этого класса показывает, что можно добавить всего 6n – 12 фиктивных операций вместо возможных $n^2 - n$ связей. Для этого класса задач список предшественников каждого элемента имеет пересечения со всеми списками предшественников других элементов.

Если n = 1 или n = 2, то фиктивных работ нет, а для каждого $n \geq 3$ добавляется 6 фиктивных по сравнению с n – 1 (см. таблицу 9). Напрашивается формула 6(n – 2) для количества фиктивных работ. В общем случае структура сети без пересечений для n предшественников приведена в таблице 10. Фиктивные работы обозначаются группой работ, заключенной в скобки, и помечены штрихами. Работы с одинаковыми предшественниками записываются слева и разделяются запятыми.

Таблица 9. Частные случаи сетей AoA, для каждого из которых в одной графе находятся a_i, а в другой – $G(a_i)$

n=1

a_1	–

n=2

a_1, a_2	–
a_3	a_2
a_4	a_1

n=3

a_1, a_2, a_3	–
a_4	a_2', a_3'
a_5	a_1', a_3''
a_6	a_1'', a_2''
a_1', a_1''	a_1
a_2', a_2''	a_2
a_3', a_3''	a_3

n=4

a_1, a_2, a_3, a_4	–
a_5	$a_2', (a_3, a_4)'$
a_6	$a_1', (a_3, a_4)''$
a_7	$(a_1, a_2)', a_4'$
a_8	$(a_1, a_2)'', a_3'$
$(a_1, a_2)', (a_1, a_2)''$	a_1'', a_2''
$(a_3, a_4)', (a_3, a_4)''$	a_3'', a_4''
a_1', a_1''	a_1
a_2', a_2''	a_2
a_3', a_3''	a_3
a_4', a_4''	a_4

Таблица 10. Общий случай структуры сети класса {*n* – *1ЭНПn*}

Количество пар фиктивных работ:		a_i	$G(a_i)$
		$a_1, \dots, a_n$	–
		a_{n+1}	$a_2', (a_3, \dots, a_n)'$
		a_{n+2}	$a_1', (a_3, \dots, a_n)''$
		⋮	⋮
	$3 \le k \le n$	a_{n+k}	$(a_1, \dots, a_{k-1})', (a_{k+1}, \dots, a_n)'$
		⋮	⋮
		a_{2n-1}	$(a_1, \dots, a_{n-2})', a_n'$
		a_{2n}	$(a_1, \dots, a_{n-2})', a_{n-1}'$
1		$(a_1, a_2)', (a_1, a_2)''$	a_1'', a_2''
		⋮	⋮
		$(a_1, \dots, a_k)', (a_1, \dots, a_k)''$	$(a_1, \dots, a_{k-1})'', a_k'$
$2(n-4)$		$(a_k, \dots, a_n)', (a_k, \dots, a_n)''$	$a_k'', (a_{k+1}, \dots, a_n)''$
		⋮	⋮
1		$(a_{n-1}, a_n)', (a_{n-1}, a_n)''$	a_{n-1}'', a_n''
		a_1', a_1''	a_1
n		⋮	⋮
		a_n', a_n''	a_n

Подтверждаем, что минимальное количество добавляемых фиктивных работ для рассмотренного класса задач равно: $2(2(n-4) + n + 2) = 6n - 12 = 6(n-2)$.

Минимальность количества фиктивных работ сетевых графиков «работы-дуги», созданных с помощью нового метода, строго не доказана, но пока и не удаётся подобрать проект, для которого бы эта минимальность не выполнялась.

Литература

1. БОГОМОЛОВ А. М. *Алгебраические основы дискретных систем*. – М.: Наука: Физматлит, 1997. – 397 с.
2. ГРИШИН А. П. *Исследование операций*. – М.: МАИ, 1975.
3. ДЫХНОВ А. Е., ПОСТОВАЛОВА И. П. *Эффективный синтез сетевой модели «Работы-Дуги* // Свидетельство об отраслевой регистрации разработки № 2687, 17.06.2003. – Москва, МОРФ, ГКЦИТ, ОФАП, 2003.
4. КРИСТОФИДЕС Н. *Теория графов. Алгоритмический подход*. – М.: Мир, 1978. – 432 с.
5. *Сетевые графики в планировании*: учеб. пособие / И. М. Разумов, Л. Д. Белова, М. И. Ипатов, А. В. Проскуряков – 3-е изд., перераб. и доп. – М.: Высш. школа, 1981. – 168 с.
6. http://www.betec.ru/secure/indexprint.php?id=11&sid=06&tid=99 (дата обращения: 14.09.2014).
7. http://www.pm-glossary.com/pmmm-glossary/1872-activity-on-arrow-network (дата обращения: 14.09.2014).

THE EFFECTIVENESS OF METHOD FOR SIMULATION OF ACTIVITY ON ARROW NETWORK ON MINIMIZATION OF THE AMOUNT OF ZERO-TIME ACTIVITY

Irina Postovalova, Chelyabinsk branch of the Financial University under the Government of the Russian Federation, Chelyabinsk, Cand.Sc., assistant professor (ira.postovalova@yandex.ru)

Abstract: *There are different types of structures of network traffic in practice: "activity-on-node" and "activity-on-directed edge" (activity on arrow network). Transfer from "activity-on-directed edge" to interfaced happens easily and without difficulties. Inverse solution is complex, because of existence of equivalent networks of "activity-*

on-directed edge" types, which differ by structure of events and zero-time activity. "Activity-on-directed edge" network does not require zero-time activity, if the lists of supporting operations agree or disjoint. Otherwise these lists are to be checked on mutual enclosure in order to reduce quantity of zero-time activity.
The effectiveness of method for reducing quantity of zero-time activity is proved on some important classes of test tasks including almost every project element.

Keywords: network model, activity on arrow network, schedule "activity-on-directed edge", zero-time activity.

Статья представлена к публикации
членом редакционной коллегии ...заполняется редактором…

Поступила в редакцию ...заполняется редактором…
Опубликована ...заполняется редактором…