УДК 681.3

АзароваО.В., ОрленкоВ.С., ХорошкоВ.А.

ОЦЕНКА ЭФФЕКТИВНОСТИ ИНФОРМАЦИОННОГО ОБЕСПЕЧЕНИЯ ЗАДАЧ КОНКУРЕНТНОЙ РАЗВЕДКИ

Понятие ***эффективности*** не имеет единого определения, что объясняется многозначностью англоязычных интерпретаций латинского слова effectus (действие): effectiveness – достижение цели независимо от затрат; efficiency – оптимизация соотношение «затраты–результаты», независимо от достижения цели; effectuality – сочетание effectiveness и efficiency. Учитывая то, что во всех приведенных интерпретациях эффективности явно фигурируют «цель» и «затраты», ***числовая оценка*** эффективности должна описываться ***не абсолютными, а сравнительными*** значениями.

В этой связи, для использования формальных методов расчета оценок эффективности это понятие целесообразно соотнести с понятием «оптимальность», то есть, нахождением условий удовлетворения заданным критериям (цель, результаты), при заданных ограничениях на изменения переменных, входящих в критерий (затраты, ресурсы). В этих терминах и определениях термин «повышение эффективности» означает ***снижение затрат*** *(использование ресурсов)*, путем ***систематического приближения к установленной цели***.

Модели оценки эффективности первоначально стали разрабатываться для использования в экономике, например, для описания темпов финансовой отдачи от инвестиций по относительной оценке «эффективность-стоимость» (cost-effectiveness modeling), при этом под термином «оптимизация эффективности» понимают ***максимизацию прибыли на единицу инвестиций***.

В отличие от экономики, ***задача оценки эффективности информацоных систем(ИС)*** стала развиваться только в последнее время, что связано с необходимостью ***минимизации реакции ИС при максимизации релевантности выдаваемых ответов*** [1].

Самым распространенным методом оценки эффективности ИС является «Оценка общей эффективности оборудования» (Overall Equipment Effectiveness, OEE), перенесенная в область ИС из области сложных промышленных многоуровневых конвейерных систем.

Вычисление оценки по методу OEE основано на перемножении трех параметров [2]:

- A – работоспособности,
- P – производительности, и
- Q – качества работы ИС.

Работоспособность ИС – время простоя $A = \square_0 / \square_e$, – где $\square_0$ операционное время, а $\square_e$ – общее затраченное время.

Производительность – отношение $P = (x_i \ \square_i) / \square_e$, – где x_i – количество элементарных запросов, а $\square_i$ – идеальный временной цикл, понимаемый как максимальная скорость обработки стандартного тестового запроса в данной ИС.

Очевидно, что производительность P не может превышать 100 %.

Качество ИС – обобщенная оценка задержек обслуживания в ИО данной ИС: $Q=x_i/R$, где – R – апостериорное число ответов, оцениваемых как релевантные (экспертная оценка).

Общий вид оценки эффективности ИС по методу OEE можно представить таким образом:

$$OEE = \frac{\square_0 \square_i x_i^{\,2}}{\square_e^{\,2} R} \qquad (1)$$

Очевидно, что для оценки эффективности в соответствии с (1) необходима система

постоянных замеров основных параметров [2].

Практика оценки эффективности реляционных баз данных отмечает ситуации, когда ответы содержали полезную информацию, но пользователю требовалось их анализировать для уточнения поиска, при этом источником задержки оказывается сам сервер базы данных, которому требуется большое время для обработки различных SQL-запросов, поэтому для повышения эффективности работы сервера пользователь должен понимать суть происходящего на сервере (т.е. суть операций между запросом и ответом).

Это потребовало создания принципиальной новой организации архитектуры информационного обеспечения (ИО), особенно с учетом инкапсуляции информационных компонентов, которая никогда не рассматривалась в стандарте SQL.

В аспекте ИО в первую очередь исследователей интересует ***релевантность*** получаемой информации, ***приведенная*** к временным затратам, т. е. отношение релевантности информации к единице времени ожидания r_j / $\square_j$, где r_j – релевантность, а $\square_j$ – время выполнения j-го запроса.

Выше было показано, что оценка технологической задержки, связанной с особенностями ПТС не является удовлетворительной, однако то же сегодня относится и к оценке релевантности. Так, на практике при оценке релевантности как правило используется упрощенная форма вычисления, в виде ***отношения количества ключевых слов k в запросе к общему количеству слов в тексте T, полученного ответа***, который может выдаваться данной ИС в различных формах – итоговой информации, справки, фрагмента документа, определенного документа, совокупности фрагментов или документов.

В этой связи считаем целесообразным введение в рассмотрение и анализ ***поисковых задержек*** в ИС, возникающих при обработке запросов к хранилищам сложно-структурированной информации и зависящим от степени релевантности всей ИС, включая задержки связанные с:

- чувствительностью ИС к запросам, в смысле семантической избирательности
- использованием методов повышение чувствительности к запросам за счёт уменьшения коэффициентов синтаксических, семантических и прагматических шумов;
- использованием методов компенсации информационных потерь за счет использования других, близких по семантике ключевых слов или ссылок;
- степенью связности семантических сетей и уровнем семантических ошибок;
- релевантностью постановок задач КР, определяющих наборы ключевых слов, мощностью тезауруса или онтологий;
- закономерностями «старения» информации и использованием методов архивирования.

В связи с этим предлагается связывать понятие ***эффективности ИО*** с задачами оптимизации, т.е. как ***оптимизацию критерия эффективности системы ИО с определенными ограничениями на входящие в критерий параметры, связанные с поисковыми задержками*** (отметим, что в такой постановке ***производительность*** ПТС может рассматриваться как внешняя константа).

Поскольку далеко не все поисковые задержки могут быть измерены непосредственно при тестировании, они могут проходить этап априорного представления в виде экспертных оценок, формализуемых четкими числовыми шкалами, нечеткими шкалами и вероятностями.

На последующих этапах эксплуатации ИО администраторы могут выработать систему апостериорных предложений по корректировке и уточнению введенных шкал и значений на основе систематического тестирования запросов или замеров с помощью специально

разработанных систем бэнчмаркинга (benchmarking).

Оценка эффективности ИО с точки зрения релевантности хранимой и используемой в ИО информации может быть формально описана как «задача о смеси» линейного программирования, информационное содержание которой в системе определяется следующим образом.

Получив от руководства задачу ***T***, понимаемую в аспекте информационного поиска как кортеж запросов ***{qj}*** ***j***-го класса к ИО КР, где ***j*** □ ***{1, 2, …, n}***, аналитик формирует и планирует проведение совокупности запросов (запросный пул), принимая во внимание, что каждый планируемый запрос ***j***-го класса должен иметь поисковую задержку не превышающую □$_j$.

Для каждого класса запросов аналитик определяет ***релевантность*** r_j, ***исходя из набора ключевых слов, заданных предметной и проблемной областями задачи T, причем суммарная временная поисковая задержка*** не должна превышать T_{max}.

В этих терминах и определениях ***задача оптимизации эффективности ИО состоит в оптимизации количества*** x_j ***запросов {qj} j-го класса, так, чтобы суммарная релевантность была максимальной при ограничении на суммарное время всех задержек***:

$$max \square r_j x_j, j \square \{1, 2, \ldots, n\} \quad (2)$$

$$\square \square_j x_j \square T_{max}, j \square \{1, 2, \ldots, n\} \quad (3$$

$$x_j \square 0, x_j - \text{целое число} \quad (4)$$

Информационное содержание задачи заключается в оптимизации априорной совокупности классов запросов, и зависящих от архитектуры ИО (т.е. фактически, от связности семантических сетей и инкапсуляции информационных компонентов в систему ИО), причем каждый из классов запросов характеризуется совокупностью поисковых задержек, удовлетворяющих заданным в системе требованиям. Таким образом, для численного решения задачи необходимы входные данные, и, прежде всего, ***список и характеристики поисковых задержек***. В этой связи авторами разработан и предлагается следующий список ***поисковых задержек***, а именно:

1. Au – достоверность, – задержка, связанная с процедурой оценки соответствия информации установленной степени доверия данному источнику.

2. Ca – синонимичность, – задержка, связанная с процедурой оценки соответствия информа-ции компонентам семантических сетей, при совпадении ключевых слов.

3. Cc – компенсаторность, – задержка, связанная с возможностью использования определен-ных фрагментов информации совместно с другими, что необходимо при низкой полноте или степени значимости информации по данному набору ключевых слов, или вместо дру-гих фрагментов при нулевой полноте или значимости ключевых слов.

4. Ci – кумулятивность, – задержка, связанная с проце́дурами присвоения одному ключевому слову нескольких индексов с целью повышения степени прагматической связности.

5. Com – полнота, – задержка, связанная с процедурой оценки соответствия информации за-данной системе ключевых слов.

6. Con – связанность, – задержка, связанная с перебором в семантических сетях всех информационных связей данного ключевого слова.

7. Efy – оперативность или своевременность, – задержка, связанная с процедурой оценки со-ответствия информации данному временному периоду («отсечение по дате»).

8. En – инкапсуляция, задержки, связанные с процедурами обращения к инкапсулированным хранилищам информации.

9. Mn – значимость, – задержка, связанная с процедурой оценки соответствия информации за-данной системе ранжирования информации, по заданным условиям приоритетности.

10. New – новизна, – задержка, связанная с процедурой выделения информации, не имеющей семантических или прагматических аналогов в семантических сетях.

11. Ns – шум, – задержка, связанная с процедурами выявления неопределенной комбинации символов или информации, несоответствующей полученным ранее достоверным данным.

12. Prs – точность, – задержка, связанная с процедурой оценки соответствия информации за-данным числовым условиям («пороговые условия»).

13. Sen – старение, – задержка, связанная с процедурой оценки соответствия информации ус-ловиям «отсечения по дате использования».

14. Sc – секретность или идентичность, – задержка, связанная с процедурой оценки соответст-вия информации праву санкционированного доступа.

15. Slc – избирательность, – задержка, связанная с процедурой оценки соответствия информа-ции данной предметной области (например, фрагменту семантической сети).

При использовании этого списка параметров поисковых задержек в численных расчетах в качестве априорной шкалы оценок предлагается выбрать единую шкалу {1-10} [2].

Следует особо отметить и подчеркнуть, что параметр ***связности*** семантической сети отличается ***двойственностью***, а именно: ***связность семантической сети увеличивает время реакции ИО на запросы, но снижение связности принципиально недопустимо, поскольку приводит к снижению релевантности***.

Поэтому для оценки эффективности ИО необходимо ***варьировать значения различных параметров*** других поисковых задержек с целью оптимизации критерия эффективности, учитывающего ***взаимоисключающие*** требования максимизации или минимизации.

Приведенная выше формализация (2-4) в виде «задачи о смеси» ставила в соответствие классам запросов априорную обобщенную релевантность r_j и поисковую задержку τ_j, не детализируя структур самих запросов, что может дать в результате полезные, но достаточно обобщенные оценки.

В этой связи предлагается ***уточненная математическая модель, в которой структура запроса учитывает связанные с ним поисковые задержки***.

Для этого введем следующие обозначения:

- $q_j = \{d_{ij}, \ldots, d_{sj}, \ldots, d_{pj}\}$ – структура запроса, как кортежа задержек d_i i-го типа;
- a_{ij} – количество задержек i-го типа в запросе j-го класса;
- b_i – ограничение снизу на количество задержек i-го типа в задаче T;
- τ_j – значение задержки запроса j-го класса (экспертная оценка, замеры, тесты);
- t_j – фиксированная задержка ПТС выполнения запроса j-го класса;

- x_j – количество запросов j-го класса в задаче T.

Задача оптимизации состоит в ***минимизации времени решения задачи с учетом уровня релевантности, установленным при решении задачи (2-4)***, а именно:

$$min \sum \tau_j(x_j), j \in \{1, 2, \ldots, n\} \quad (5)$$

$$\sum a_{ij} x_j \le b_i, \; i \in \{1, 2, \ldots, m\}, \; x_j \ge 0 \quad (6)$$

где

$$\tau_j(x_j) = \begin{cases} 0, & \text{при } x_j = 0 \\ \tau_j x_j + t_j, & \text{при } x_j > 0 \end{cases} \quad (7)$$

При введении ***верхней границы*** $x_j \le k_j$, $j \in \{1, 2, \ldots, n\}$ задача примет вид:

$$min \sum (\tau_j x_j + t_j y_j), j \in \{1, 2, \ldots, n\}$$

$$y_j = \begin{cases} 0 \text{ при } 0 \le x_j \le k_j \\ 1 \text{ в противном случае} \end{cases}$$

Как было показано, для постановки задачи оптимизации эффективности ИО и ее последующей формализации необходимо априорное введение на первом этапе лингвистических переменных основных параметров релевантности и шкал их оценок. При этом следует учитывать, что значимость различных классов запросов может варьировать от задачи к задаче, формирование эффективного ИО, отвечающего требованию устанавливаемой релевантности совокупности запросов, укладывающейся в допуски поисковых задержек, может дать определенный эффект, выражающийся в повышении производительности системы в целом.

Для решения этой задачи и был разработан алгоритм оптимального выбора источника информации на сети в виде дерева.

Пусть задано дерево $H = (X, V)$, каждой вершине которого приписано число $q(x) > 0$.

Обозначим через $X_{v,x}$ множество вершин дерева H, обладающих свойством $y \in X_{v,x}$ тогда и только тогда, когда в H простая цепь $C(x,y)$, соединяющая x и y, содержит ребро v. Тогда величина

$$\alpha_x(v) = \sum_{y \in X_{v,x}} q(y) \quad (8)$$

будет потоком по ребру v.

Рассмотрим функционал

$$F(x) = \sum_{v \in V} \psi_v(\alpha_x(v)), \quad (9)$$

где ψ_v – положительные возрастающие функции. Требуется найти такую величину $x^* \in X$, для которой функционал (9) достигает своего минимального значения.

Пусть $H^1_k = (X^1_k, V^1_k)$ и $H^2_k = (X^2_k, V^2_k)$ – подграфы, на которые распадается дерево H при удалении ребра $v_k = (x_k, y_k)$, причем $x_k \in X^1_k$, а $y_k \in X^2_k$.

Тогда $\alpha y_k(v_k)$ и $\alpha x_k(v_k)$ будут суммами весов вершин этих подграфов.

Рассмотрим вариацию $\Delta(x_k, y_k)$ функционала $F(x)$ при переходе от вершины x_k к смежной вершине y_k:

$$\Delta(x_k, y_k) = F(x_k) - F(y_k) = \sum_{v \in V} \psi_v(\alpha x_k(v)) - \sum_{v \in V} \psi_v(\alpha y_k(v)) =$$

$$= \sum_{v \in V^1_k} \psi_v(\alpha x_k(v)) + \psi v_k(\alpha x_k(v_k)) + \sum_{v \in V^2_k} \psi_v(\alpha x_k(v)) - \sum_{v \in V^1_k} \psi_v(\alpha y_k(v)) - \psi v_k(\alpha y_k(v_k)) -$$

$$- \sum_{v \in V^2_k} \psi_v(\alpha y_k(v)) = \psi v_k(\alpha x_k(v_k)) - \psi v_k(\alpha y_k(v_k)) \quad (10)$$

Знак вариации $\Delta(x_k, y_k)$ совпадает со знаком выражения $\alpha x_k(v_k) - 1/2Q$, где

$$Q = \sum_{y \in X} q(y) \qquad (11)$$

Действительно, поскольку ψ_v – возрастающие функции для всех $v \in V$, то знак $\Delta(x_k, y_k)$ совпадает со знаком выражения $\alpha x_k(v_k) - \alpha y_k(v_k)$. Но так как $\alpha x_k(v_k) + \alpha y_k(v_k) = Q$, то знак $\Delta(x_k, y_k)$совпадает с $\alpha x_k(v_k) - 1/2Q$.

Будем говорить, что на простой цепи $C(x,y)$ дерева H справедливо, что $v_1 > v_2$, если $v_1 \neq v_2$ и, следуя из вершины x в y, встречаем ребро v_1, а затем ребро v_2.

В [3] доказано, что если на простой цепи $C(x,y)$ дерева H имеет место $v_1 > v_2$ ($v_1 = (x_1, y_1)$, $v_2 = (x_2, y_2)$)и если $\Delta(x_1, y_1) \leq 0$, то $\Delta(x_2, y_2) < 0$. Действительно, так как на простой цепи $C(x,y)$ дерева H выполняется $v_1 > v_2$, то любая вершина подграфа H^2_2 принадлежит и подграфу H^2_1. Вместе с тем, вершина x_2 принадлежит H^2_1, но не принадлежит подграфу H^2_2. Поскольку для любого $y \in X$ справедливо $q(y) > 0$, то имеем $\alpha y_2(v_2) > \alpha y_1(v_1)$ и $\alpha x_2(v_2) < \alpha x_1(v_1)$, а, следовательно,

$$\psi v_2(\alpha y_2(v_2)) > \psi v_1(\alpha y_1(v_1)) \text{ и } \psi v_2(\alpha x_2(v_2)) < \psi v_1(\alpha x_1(v_1)).$$

Но тогда

$$\psi v_2(\alpha x_2(v_2)) - \psi v_2(\alpha y_2(v_2)) < \psi v_1(\alpha x_1(v_1)) - \psi v_1(\alpha y_1(v_1)) \leq 0,$$

то есть

$$\Delta(x_2, y_2) < 0.$$

А это значит, что локальный оптимум совпадает с глобальным и что множество X^* вершин x^* дерева H, которые минимизируют функционал (9), состоит не более чем из двух вершин, и если их две, то они смежны. Пусть $V(x)$ – множество ребер дерева H, инцидентных вершин $x \in X$.

ТЕОРЕМА 1. Вершина дерева минимизирует функционал (9) тогда и только тогда, когда выполнено условие

$$\max_{v \in V(x^*)} \alpha x^*(v) \leq \tfrac{1}{2} Q \qquad (12)$$

Доказательство. Достаточность. Пусть x^* – вершина дерева H, для которой выполняется условие (12). А это значит, что $\alpha x^*(v) \leq \frac{1}{2} Q$ для любого $v \in V(x^*)$. На основе изложенного выше следует, что вдоль любой простой цепи типа $C(x^*,x)$ функционал (8) будет увеличиваться, а, следовательно,

$$F(x^*) = \min_{x \in X} F(x) \qquad (13)$$

Необходимость. Пусть x^* – вершина дерева H, для которой выполняется условие (13).

Тогда вариация $\Delta(x^*, y) \leq 0$ для любого $y \in X^*$, то есть

$$\max_{v \in V(x^*)} \alpha x^*(v) \leq \tfrac{1}{2} Q$$

СЛЕДСТВИЕ 1. Оптимальное местоположение источника на сети в виде дерева не зависит от конкретного вида возрастающих функций $\psi_v(\alpha(v))$.

ЗАМЕЧАНИЕ 1. Если для любого $v \in V$ взять $\psi_v(\alpha(v)) = l(v) * \alpha(v)$, где $\alpha(v)$ – поток по ребру v, а $l(v)$ – длина ребра v, то

$$F(x) = \sum_{v \in V} \psi_v(\alpha_x(v)) = \sum_{v \in V} l(v) * \sum_{y \in X_{v,x}} q(y) = \sum_{x \in X} q(y) * d(x,y), \text{ где}$$

$$d(x,y) = \sum_{v \in C(x,y)} l(v)$$

Таким образом, задача об оптимальном выборе источника на сети в виде дерева свелась к задаче нахождения медианы дерева. В дальнейшем будем считать, что $l(v)=1$, то есть $\psi_v(\alpha(v))=\alpha(v)$ для любого $v \in V$.

Рассмотрим задачу нахождения медианы дерева. Дерево H может быть отображено в линейном пространстве $R^{n-1}{}_1$ следующим образом [4, 5]. Пусть I^{n-1} – единичный куб пространства $R^{n-1}{}_1$, то есть $I^{n-1} = \{ Z = (z^1, z^2, \ldots, z^k, \ldots, z^{n-1}) \mid 0 \le z^k \le 1 \}$. Фиксируем произвольную вершину x_1 дерева H. Теперь каждой вершине x_i дерева H поставим в соответствие точку z_i в пространстве $R^{n-1}{}_1$ с координатами

$$z_i = (z^1{}_i, z^2{}_i, \ldots, z^k{}_i, \ldots, z^{n-1}{}_i),\ i = 1, 2, \ldots, n. \qquad (14)$$

Где $z^k{}_i$ равно нулю либо единице в зависимости от того, входит или не входит в простую цепь $C(x_1,x_i)$ ребро $v_k \in V$, $k=1, 2, \ldots, n-1$. Если x_i и x_j – смежные вершины дерева H, то соединяем отрезком (ребром) точки z_i и z_j.

Получаем дерево H' в пространстве $R^{n-1}{}_1$. Из (14) следует, что различным вершинам дерева H соответствуют при этом различные вершины куба I^{n-1}, причем смежным вершинам дерева H соответствуют смежные вершины куба I^{n-1} . Поэтому каждому ребру $v_k \in V$ поставлено в соответствие то ребро w_k куба I^{n-1}, концы которого суть образы вершин ребра v_k.

Следовательно, отображение реализует дерево H в дерево H' на кубе I^{n-1} изоморфным образом. Заметим, что если z_i и z_j – две смежные вершины дерева H' и цепь $C(z_1,z_i)$ содержит вершину z_j, то вектор $z_i - z_j$ пространства $R^{n-1}{}_1$ согласно (14) есть единичный вектор этого пространства, а, отсюда, как нетрудно заметить, следует, что ребра $w_1, w_2, \ldots, w_{n-1}$ дерева H' имеют разные координатные направления. Каждой вершине z_i дерева H' припишем тот же вес, что и вершине x_i дерева H. Итак, задача нахождения медианы дерева H сведена к задаче о нахождении медианы в пространстве $R^{n-1}{}_1$, когда система точек $S = \{z_1, z_2, \ldots, z_n\}$ состоит из вершин единичного куба I^{n-1}. Множество решений последней задачи представляет собой [5] либо нульмерную, либо одномерную грань куба I^{n-1}. Вершины этой грани I^t $(t = 0;1)$ принадлежат дереву H'.

На основе доказанного выше можно предположить следующие алгоритмы нахождения медианы на дереве.

Алгоритм 1:

1) берем произвольную вершину x_1 дерева H и составляем матрицу $R^v{}_x = \|z^k{}_i\|$, строчки которой соответствуют вершинам дерева H, а столбцы – ребрам, причем $z^k{}_i = 1$, если цепь дерева H, ведущая из x_1 в x_i, содержит ребро v_k, а в противном случае $z^k{}_i = 0$;

2) Вычисляем скалярное произведение (Z^k, P) векторов Z^k, P, где $Z = (z^k{}_1, z^k{}_2, \ldots, z^k{}_n)$ – столбец матрицы $R^v{}_x$, а $P = (q(x_1), q(x_2), \ldots, q(x_n))$ и соответственно числа $S^k = (Z^k, P) - \frac{1}{2} Q$, где $k = 1, 2, \ldots, n-1$;

3) положим

$$x^k{}_* = \begin{cases} 0, \text{ если } S^k < 0, \\ 1, \text{ если } S^k > 0, \\ 0 \text{ или } 1 \text{ (безразлично), если } S^k = 0. \end{cases}$$

и рассмотрим вектор $x^* = (x^1{}_*, x^2{}_*, \ldots, x^{n-1}{}_*)$;

4) выявляем строку матрицы $R^v{}_x$, совпадающую с вектором $x^* = (x^1{}_*, x^2{}_*, \ldots, x^{n-1}{}_*)$.

Так как деревья H и H' изоморфны, то вектор $x^* = (x^1{}_*, x^2{}_*, \ldots, x^{n-1}{}_*)$ обязательно совпадет с одной из строк матрицы $R^v{}_x$;

вершина x^*, соответствующая этой строке, является оптимальной в дереве H;

существует не более одного столбца, для которого $S^k = 0$, так как не более двух оптимальных вершин (медиан), в дереве H, причем если их две, то они смежны.

Таким образом, алгоритм отыскания медианы состоит в определении вектора $x^* = (x^1{}_*, x^2{}_*, \ldots, x^{n-1}{}_*)$, то есть знаков величин $S^k = (Z^k, P) - \frac{1}{2} Q$ для всех ребер дерева H, а затем в выявлении строк матрицы $R^v{}_x$, которые совпадают с вектором $(x^1{}_*, x^2{}_*, \ldots, x^{n-1}{}_*)$.

При таком алгоритме, количество операций соответствует порядку $O(n^2)$. Можно предположить некоторую модификацию этого алгоритма. Заметим, что в k -том столбце

матрицы R^v_x единицами отмечены те вершины, которые достижимы из вершины x_1 дерева H при прохождении по ребру v_k.

Пусть $v_k = (x_k, y_k)$ – произвольное ребро дерева H. Как и раньше, через H^1_k и H^2_k обозначим подграфы, на которые распадается дерево H при удалении ребра v_k, а через $q(H^1_k)$ и $q(H^2_k)$ – значения весов вершин этих подграфов.

ОПРЕДЕЛЕНИЕ 1. Операцией v_k -усечения дерева H называется замена дерева H на H^1_k с присвоением вершине x_k нового веса $q'(x_k) = q(x_k) + q(H^2_k)$, если $q(H^1_k) > q(H^2_k)$, или наоборот замена на H^2_k, с присвоением вершине y_k нового веса $q'(y_k) = q(y_k) + q(H^1_k)$, если $q(H^1_k) < q(H^2_k)$.

Алгоритм 2:

1) проверяем, существует ли вершина x^* дерева H такая, что $q(x^*) \geq ½ Q$. Если да, то переходим к пункту 6, в противном случае – к п.2.;

2) берем произвольную вершину x_i и произвольное ребро v_k дерева H. Находим множество вершин X^2_k, которое достижимо из выбранной вершины x_1 при прохождении по ребру v_k, вычисляем $q(H^2_k)$ и $q(H^1_k) = Q - q(H^2_k)$;

3) проверяем $q(H^2_k) \geq q(H^1_k)$. Если да, то переходим к пункту 4, в противном случае – к п.5.;

4) применяя операцию v_k -усечения дерева H, заменяем дерево H на H^2_k и переходим к п.1.;

5) применяя операцию v_k -усечения дерева H, заменяем дерево H на H^1_k и переходим к п.1.

6) печатаем номер вершины, являющейся решением.

ЗАМЕЧАНИЕ 2. Если ребра v_k, использованные в модифицированном алгоритме, образуют цепь с началом в вершине x_1 дерева H, то количество итераций не превышает d, где d – длина диаметра дерева H. На этом основывается третий алгоритм нахождения медианы в дереве [3].

Если же ребра v_k, использованные на каждой итерации модифицированного алгоритма, инцидентны висячим вершинам деревьев, то мы получим четвертый алгоритм нахождения медиан деревьев.

Алгоритм 3:

1) проверяем, существует ли вершина x^* дерева H такая, что $q(x^*) \geq ½ Q$. Если да, то переходим к пункту 6, в противном случае – к п.2.;

2) берем произвольную вершину x_1 дерева H;

3) вычисляем вариацию $\Delta(x, y)$ для всех ребер, инцидентных вершине x_1. Находим

$$\Delta(x_1, y_1) = \max_{y \in O(x_1)} \Delta(x_1, y)$$

и проверяем $\Delta(x_1, y_1) \geq 0$. Если да, то переходим к п.4., в противном случае – к п.6.

4) далее проверяем $\Delta(x_1, y_1) = 0$. Если да, то x_1 и y_1 – медианы дерева H. Тогда переходим к п.6., в противном случае – к п.5.;

5) заменяем x_1 на y_1 и переходим к п.3.;

6) печатаем номер вершины, являющейся решением.

Данный алгоритм есть аналог метода градиентного спуска, осуществляемого по единственно простой цепи $C(x_1,x^*)$, которая идет от первоначально выбранной вершины x_1 к оптимальной вершине x^* дерева H.

Как видно из алгоритма, вычисление вариации $\Delta(x, y)$ необходимо только для ребер, инцидентных вершинам x, принадлежащим простой цепи $C(x_1,x^*)$.

ОПРЕДЕЛЕНИЕ 2. Операцией усечения дерева H называется [6] удаление его висячих вершин вместе с инцидентными им ребрами и с одновременным прибавлением весов удаляемых вершин к весам соответствующих смежных им вершин, соблюдая следующие правила:

а) операция усечения применяется лишь к деревьям, имеющим больше двух вершин;

б) удаляются все висячие вершины дерева H, если среди них имеются, по крайней

мере, две вершины с одинаковыми весами;

в) если среди висячих вершин дерева H имеется лишь одна вершина с максимальным весом, то удаляются все остальные вершины.

Заметим, что, применяя операцию усечения к дереву H, получим новое дерево H_1. После конечного числа этапов применения операции усечения к деревьям $H, H_1, H_2, \ldots, H_{p-1}$ получим либо одну вершину (вырожденное дерево), либо дерево H_p с двумя вершинами.

Алгоритм 4:

1) проверяем, существует ли вершина x^* дерева H такая, что $q(x^*) \geq \frac{1}{2} Q$. Если да, то переходим к пункту 5, в противном случае – к п.2.;

2) находим висячие вершины дерева H;

3) применяем операцию усечения дерева H;

4) в результате выполнения п. 2 и п. 3, получаем новое дерево H_1. Заменяем H на H_1 и переходим к п.1;

5) печатаем номер вершины x^*, являющейся оптимальным решением.

Характерным для данного алгоритма является то, что он позволяет найти $q(H^2_k)$ и $q(H^1_k)$ для любого $k = 1, 2, \ldots, n-1$, выполнив наименьшее число арифметических операций по суммированию весов.

При оценке работы (трудоемкости) алгоритма нужно, однако, учесть, что на каждом этапе мы должны найти висячие вершины дерева.

Последний алгоритм одновременно позволяет решить и другие задачи [7], родственные задаче о медиане дерева.

Рассмотрим некоторые задачи, родственные задаче о медиане дерева.

ЗАДАЧА 1. Найти вершину x^0 дерева H, в которой функционал (9) достигает максимального значения.

Обозначим через X^0 множество вершин x^0 дерева H, являющихся решением задачи 1.

ТЕОРЕМА 2. Множество X^0 состоит только из висячих вершин дерева H.

Доказательство. Допустим вначале, что x^* – единственная вершина дерева H, минимизирующая функционал (8). Тогда на основе теоремы 1 на любой простой цепи $C(x^*,y)$ дерева H функционал $F(x)$ строго возрастает и тем самым его максимальное значение нужно искать среди висячих вершин дерева H.

Пусть теперь x^* и y^* – две вершины дерева H, в каждой из которых функционал (9) достигает своего наименьшего значения. Как уже отмечалось раньше, эти две вершины должны быть смежными.

Рассмотрим простые цепи $C(x^*,y)$, $C(y^*,y)$, не проходящие через вершины x^* и y^* соответственно. Вдоль этих цепей функционал $F(x)$ строго возрастает. Таким образом, его максимум во всех случаях нужно искать среди висячих вершин дерева H, что и требовалось доказать.

ЗАМЕЧАНИЕ 3. Как известно [7], при решении задач размещения пунктов производства учитываются не только транспортные расходы, но и затраты на производство соответствующего количества продукта в заданном пункте производства. Поэтому представляет интерес

ЗАДАЧА 2. Найти вершину x' дерева H, в которой

$$f(x) = Q * r(x) + \sum_{v \in V} \psi_v(\alpha_x(v)), \qquad (15)$$

достигает минимального значения, где $r(x)$ – неотрицательная вещественная функция, определенная на X (например, затраты на производство продукции в вершине x).

Множество вершин x' дерева H, в котором функционал (15) достигает минимального значения, обозначим через X'.

ЗАМЕЧАНИЕ 4. Множество X' может совпадать с множеством X.

Заметим, что в результате последовательного применения операции усечения дерева, каждой вершине $y \in X$ мы ставим в соответствие «потенциал» $\alpha(y)$, равный потоку $\alpha_{x^*}(v)$, где v – последнее ребро простой цепи $C(x^*,y)$, а x^* – медианная вершина дерева H.

Теперь можно изложить **алгоритм 5**, который позволяет в произвольном дереве найти множества X^*, X^0 и X':

1) проверяем выполнение условия $n > 2$. В случае его выполнения переходим к пункту 2, в случае невыполнения – к п.5.;

2) находим висячие вершины дерева H;

3) применяем операцию усечения дерева H;

4) в результате выполнения п. 2 и п. 3, получаем новое дерево H_1 с количеством вершин $n_1 < n$. Заменяем H на H_1 и число n на n_1. Переходим к п.1;

5) после конечного числа итераций всем вершинам дерева H будут присвоены «потенциалы» $\alpha(y)$; таким образом, получим дерево H_p, которое содержит одну или две вершины. Если H_p содержит одну вершину, то переходим к п. 6, в противном случае – в пункту 7;

6) печатаем номер этой вершины (единственная медиана дерева H) и переходим к п. 10;

7) пусть x_1 и x_2 – вершины дерева H, а $\alpha(x_1)$ и $\alpha(x_2)$ – соответствующие «потенциалы». Если $\alpha(x_1) = \alpha(x_2)$ то переходим к п. 8, в противном случае – к п. 9;

8) печатаем номера этих вершин (множество медиан дерева H) и переходим к пункту 10;

9) выбираем вершину дерева H с максимальным полученным потенциалом. Печатаем номер этой вершины (единственная медиана дерева H) и переходим к п. 10;

10) найденную медианную вершину дерева H (если их две, то любую из них) выбираем в качестве корня для H и ориентируем ребра в направлении от нее. В результате получим ордерево $\vec{H} = (x, V)$;

11) для каждой дуги $v_k = (x_k, y_k)$ ордерева $\vec{H}$ определяем длину $l(v_k) = \psi v_k(Q - \alpha_{x^*}(v_k)) - \psi v_k(\alpha_{x^*}(v_k))$. Так как x^* – медианная вершина дерева H, то все $l(v_k) \geq 0$ (заметим, что только для одной дуги $l(v_k) = 0$ и то лишь в случае, когда имеются две медианные вершины);

12) применяя на ордереве $\vec{H}$ с вычисленными длинами дуг $l(v_k)$ алгоритм кратчайших путей от вершины x^* до остальных вершин находим висячие вершины, наиболее удаленные от x^*; они образуют множество X^0. Печатаем номера этих вершин;

13) для каждой дуги v_k ордерева $\vec{H}$ определяем число $\Delta r(v_k) = r(y_k) - r(x_k)$, которое может быть как положительным, так и отрицательным;

14) определяем для каждой дуги $v_k \in V$ «длину» $L(v_k) = l(v_k) + \Delta r(v_k)$. Заметим, что числа $L(v_k)$ могут быть как положительными, так и отрицательными;

15) находим в ордереве $\vec{H}$ с вычисленными «длинами» дуг наиболее удаленные вершины от медианной вершины x^*. Они и образуют множество вершин X' дерева H.

Применяя операцию усечения дерева (выполняя пункты 1-10 алгоритма), получаем H_1, а затем H_2.

Поскольку в полученном дереве H_2 $\alpha(x_6) > \alpha(x_4)$, x_6 – единственная медианная вершина дерева H, т.е. $X^* = \{x_6\}$. Если источник разместить в вершине x_6, то в результате применения операции для каждого ребра v_k получится поток $\alpha(v_k)$.

Выполним далее пункты 11-15 алгоритма, получаем $X^o = \{x_2\}$ и $\overline{X} = \{x_3\}$.

Проведем исследование вопроса устойчивости [8,9] оптимальной медианной вершины дерева.

Рассмотрим в связи с этим , параметрическую задачу: найти множество вершин x*(t) дерева ***H=(X,V)***, в котором функционал

$$F_t(x) = \sum_{v \in} \psi_v\left(\alpha^t_x(v)\right) \qquad \textbf{(16)}$$

достигает своего наименьшего значения. Здесь $\alpha^t_x(v) = \sum_{y \in Xv,x}\left(a(y)+b(y)\right)$, где *a(y), b(y)* – вещественные числа, а ***t*** – параметр.

Таким образом, когда спрос ***q(x)*** линейно зависит от параметра ***t***, т.е. ***q(x)=a(x)+tb(x)***, мы получаем задачу о нахождении медианы ***x*(t)*** дерева *H*.

Как и ранее находим сначала интервал определенности ***(A, B)***. Для этого достаточно решить систему неравенств

$$a(x)+tb(x) \geq 0 \; x \in X \qquad \textbf{(17)}$$

Множеством оптимальности ***(t,t)*** для вершины ***x**** дерева ***H*** называется множество значений параметра ***t***, для которого наименьшее значение функционала (16) достигается в вершине ***x****. Следовательно, если разбить интервал определенности ***(A, B)*** на множество оптимальности. То тем самым решим поставленную параметрическую задачу.

На основании изложенного предлагается простой алгоритм решения параметрической задачи на дереве.

Решая систему линейных неравенств (17) относительно параметра ***t***, получаем интервал определенности ***(A, B)***. Далее берем конкретное значение параметра ***t=c***, где ***A<c<B***, и находим медианную вершину x_1^* дерева ***H***. На основании этого строим систему линейных неравенств относительно ***t***:

$$\alpha_{x_1^*}(v) \leq \frac{1}{2}\left(\sum_{x \in X} a(x) + t\sum_{x \in X} b(x)\right), v \in V(x_1^*). \qquad (18)$$

Здесь $V(x_1^*)$ – множество ребер дерева ***H*** инцидентных вершине x_1^*.

Решая (18) и учитывая ***A<t<B***, находим интервал $(\underline{t_1}, t_1)$, т.е. множество оптимальности медианой вершины x_1^*.

Если $A<\underline{t_1}<\bar{t}_2<B$, то это означает, что при $t=\underline{t_1}$ в системе (18) одно из неравенств выполняется как равенство. Следовательно, в этом случае кроме медианой вершины x_1^* дерева ***H*** имеется еще и другая медианная вершина x_2^*. Относительно x_2^* строим систему линейных неравенств

$$\alpha_{x_2^*}(v) \leq \frac{1}{2}\left(\sum_{x \in X} a(x) + t\sum_{x \in X} b(x)\right), v \in V(x_2^*)/(x_1^*, x_2^*) \quad (19)$$

при решении которой с учетом ***A<t<B***, находим интервал $(\underline{t_2}, t_2=\underline{t_1})$ – множество оптимальности медианой вершины x_2^*. Эта операция должна повторяться до тех пор, пока не получим значение ***t=A***, а затем ***t=B***. алгоритм закончен, так как каждый раз переходит от одной вершины дерева ***H*** к другой.

Заметим, что если граф ***H***-дерево, то либо всему интервалу определенности соответствует одна или две смежные медианные вершины, либо интервал определенности разбивается на несколько интервалов, каждому из которых соответствует лишь одна медианная вершина и при переходе в соседний интервал она смещается в смежную вершину дерева.

Применение предложенных алгоритмов оптимального выбора источника информации на сети в виде дерева позволяет достаточно просто, быстро, и с заданной точностью и эффективностью решать задачи ИО КР.

Список литературы:

1. *Азарова О.В.* Анализ рисков системе конкурентной разведки. / Сборник научных трудов НАУ «Защита информации» - К.: Издательство НАУ 2004. с.34-45.

2. *Андреев В.И., Козлов В.С., Хорошко В.А.* Количественная оценка защищенности технических объектов с учетом их функционирования. / Захист інформації №2, 2004. с.47-51.

3. *Кальманс А.К.* О выборе оптимальной вершины в графе. / В кн. «Исследования по дискретной математике». – М.: Наука, 1989. с. 151-158.

4. *Замбицкий Д.К., Солтан П.С.* Об одной экстремальной задаче на дереве. / В кн. «Математические методы решения экономических задач». Вып. 11. – М.: Наука, 1989. с. 102-107.

5. *Замбицкий Д.К.* Относительно одной экстремальной задачи на графе. / В кн. «Прикладная математика и программирование». – Кишинев: РИО АК МССР, 1988. с. 18-27.

6. *Духовный М.А.* Об одной оптимальной задаче теории графов. / В кн. «Математические заметки». Вып. 3. – М.: Наука, 1981, 10. с. 355-359.

7. *Хорошко В.О., Кудінов В.А.* методичний підхід формалізації задачі оцінювання ефективності системи захисту інформаційної системи ОВС України. / Захист інформації, №4, 2004. с. 11-18.

8. *Сибирский В.К.* Параметрическая задача Штейнера на графах. / Известия АН МССР. Сер. физ-техн. и математ. наук, №3, 1986. с. 22-25.

9. *Гольштейн Е.Г., Юдин Д.Б* Задачи линейного программирования транспортного типа. – М.: Наука, 1989. 254с.

Поступила 20.10.2004г.

УДК 681.188; 004.056.5; 004.421.2:517.518.26

Васильцов І.В., Васильків Л.О.

СТІЙКІСТЬ СУЧАСНИХ АЛГОРИТМІВ МОДУЛЯРНОГО ЕКСПОНЕНЦІЮВАННЯ ДО ЧАСОВОГО АНАЛІЗУ

Вступ

На сучасному етапі розвитку криптографії існує багато різноманітних методів та засобів захисту інформації. Поширеними та популярними стали методи захисту, які базуються на асиметричній криптографії, оскільки її застосування дозволило вирішити задачу розподілу ключів та електронного цифрового підпису [1-5].

Асиметричні криптосистеми на сьогоднішній день досліджені менше, ніж симетричні, оскільки концепція їх побудови була запропонована відносно недавно.

Вагомою причиною, яка не дозволяє використовувати асиметричні алгоритми шифрування для захисту інформації в системах передачі даних – це низька швидкодія виконання основних математичних процедур, які використовуються для шифрування та дешифрування. Цей факт особливо характерний для реалізацій алгоритмів на пристроях із невеликими обчислювальними можливостями.

Одним із шляхів вирішення цієї проблеми є зменшення розмірності параметрів системи, але це може привести до зменшення стійкості системи. Іншою можливістю є застосування алгоритмів, призначених для ефективного здійснення основних математичних обчислень, які використовуються в асиметричних криптосистемах [2,6,7]. При цьому