

УДК 004.65

Адаптивный нечёткий алгоритм кэширования для прокси-серверов

А. И. Жуков

(Донской государственный технический университет)

Рассматривается проблема увеличения эффективности кэш-систем за счёт использования методов искусственного интеллекта. Приведена формализованная постановка задачи повышения эффективности систем кэширования с использованием теоретико-множественного математического аппарата. Рассмотрена технология применения системы нечёткого вывода для определения объекта-жертвы, реализующая вычисление кэш-рейтинга объектов, сохранённых в кэш-памяти. В качестве одной из базовых характеристик web-ресурса, подаваемых на вход системы нечёткого вывода, впервые предложено использовать его пространственную локальность, определение которой основано на унифицированном идентификаторе web-ресурса URI. Определён способ выполнения адаптации нечёткой системы посредством синтеза лингвистической модели с использованием метода нечёткой кластеризации, а также приведены результаты экспериментального исследования разработанного алгоритма на имитационном стенде для трасс, полученных с использованием кэширующего прокси-сервера Squid.

Ключевые слова: кэш-система, система нечёткого вывода, нечёткая кластеризация, адаптивная система кэширования.

Введение. Растущая популярность web-технологий и Интернета позволили пользователям всего мира обмениваться большим числом данных, среди которых: страницы с электронным контентом, фото и видео материалы, документы различных форматов и размеров. Ещё в 1999 году размер файла в пятьдесят мегабайт назывался исследователями web-систем и прокси-серверов «экстремально большим» («extremely large object») [1], а сегодня ежедневно пользователи имеют возможность скачивать с различных серверов фильмы, размер каждого из которых составляет несколько гигабайт. Увеличение быстродействия web-систем, несомненно, связано с увеличением производительности аппаратного обеспечения, ростом числа серверов в глобальной паутине, появлением новых web-технологий и совершенствованием протоколов передачи данных. Однако человеческая природа постоянно требует всё более скоростных методов доступа к информации с меньшими временными задержками. В этом смысле использование кэширования в web-системах является одним из способов увеличения производительности, который, благодаря универсальности, может быть применён на различных уровнях функционирования web-систем [2, 3].

Одно из важнейших звеньев в реализации запроса конечного пользователя к оригинальному web-серверу с данными — это кэширующий прокси-сервер, предназначенный для выполнения косвенных запросов клиентов к другим серверам или сетевым службам. Применение кэширования на прокси-сервере позволяет [4]:

- увеличить производительность серверов, функционирующих в рамках сетевой информационной системы, так как кэш-система будет удовлетворять часть запросов и в результате сервер сможет обслуживать большее число запросов в единицу времени;
- уменьшить среднее значение времени ожидания пользователем запрошенных данных;
- уменьшить нагрузку на локальные и глобальные сети за счёт применения кэширования на локальных уровнях;
- перераспределить нагрузку между различными центрами обработки информации единой системы за счёт перенаправления кэш-запросов на системы с меньшей загрузкой.

Ключевым звеном системы кэширования является алгоритм кэширования, который на базе явно или неявно представленного кэш-рейтинга объектов системы определяет, какие объекты

должны быть вытеснены из кэш-памяти при её переполнении. Вытесняемый из кэш-памяти объект, т. е. объект с минимальным рейтингом, называют *жертвой*. Определение жертвы — задача стратегии замещения.

В последнее время наблюдается рост популярности применения известных технологий и методов искусственного интеллекта в различных прикладных задачах [5]. В работах [6, 7] представлено семейство *нечётких алгоритмов замещения* (англ. *fuzzy algorithm replacement*), разработанных для применения в системах web-кэширования. Несмотря на нашедшую отражение в этих работах эффективность нечётких стратегий, представленные алгоритмы имеют существенные недостатки:

- необходимость предварительного анализа потока запросов к кэш-системе для формирования базы правил;
- невозможность адаптации кэш-системы к изменениям законов распределения вероятностей доступа к объектам, происходящим в трассе.

В данной работе определён метод разработки адаптивной системы кэширования на базе нечёткой стратегии замещения с использованием системы нечёткого вывода (CHV), а также модуля адаптации на базе нечёткой кластеризации для синтеза лингвистической модели.

Постановка задачи. Сформулируем математическую постановку решаемой научной задачи. Для этого введём в рассмотрение некоторые понятия.

Пусть R — множество алгоритмов кэширования (англ. *replacement algorithm*):

$$R = \{A_1, A_2, \dots, A_r\}, \quad (1)$$

где A_i — представляет i -й алгоритм из множества R , определённый в соответствии с математической моделью абстрактной одноуровневой кэш-системы [8]; r — мощность множества R , т. е. число алгоритмов принадлежащих множеству ($r = |R|$).

Пусть $\omega(N, t, T)$ — случайная функция, результатом которой является случайная трасса ω , соответствующая модели потока запросов [8] и составленная из T идентификаторов r_i объектов системы ($r_i \in N$), полученных начиная с момента времени t :

$$\omega = (r_1, r_2, \dots, r_T). \quad (2)$$

В общем случае ω является элементарным исходом случайного процесса и принадлежит пространству элементарных событий Ω_T ($\omega \in \Omega_T$), которое включено в T -ую Декартову степень множества идентификаторов объектов системы:

$$\Omega_T \subseteq N^T, \quad (3)$$

где N — множество идентификаторов объектов системы.

Введём в рассмотрение отображение ϕ , определяющее случайное положительное рациональное число ρ , которое представляет значение рейтинга кэш-попаданий (англ. *hit-ratio*), полученное для кэш-памяти объёма M , с использованием алгоритма кэширования A_i и случайной трассы длиной T , полученной для множества идентификаторов объектов системы N , начиная с момента времени t :

$$\phi(A_i, M, \omega(N, t, T)) = \rho, \quad (4)$$

$$\phi: R \times \mathbb{N} \times \Omega_T \rightarrow \mathbb{R}_+, \quad (5)$$

где R — множество алгоритмов кэширования (1); Ω_T — множество случайных трасс длины T ; $\mathbb{N}$ — множество натуральных чисел; $\mathbb{R}_+$ — множество положительных рациональных чисел.

В общем случае требуется реализовать такой алгоритм кэширования A_i , который в среднем обеспечивает наибольший рейтинг кэш-попаданий для заданного объёма кэш-памяти, множества

объектов системы и их идентификаторов, на всех трассах длины T , т. е. решить задачу стохастического программирования с моделью ожидаемого среднего значения (*expected value model* — EVM) [9]:

$$\max_{A \in R} E(\phi(A, M, \varpi(N, t, T))), \quad (6)$$

где $E(\bullet)$ — операция нахождения среднего значения.

Адаптивный нечёткий алгоритм кэширования. Вычисление кэш-рейтингов объектов в системе кэширования на базе нечёткой логики основано на системе нечёткого вывода (СНВ), структура которой представлена ниже (рис. 1).



Рис. 1. Структурная схема системы нечёткого вывода

На схеме представлены следующие модули СНВ:

- блок фаззификации (англ. *fuzzification*), предназначенный для перевода параметров расчёта, представленных вектором чётких чисел $\bar{x}$, в нечёткие множества A_i для последующего использования в блоке вывода;
- блок дефаззификации, применяемый для перевода полученных нечётких множеств B_j в вектор чётких значений выходных данных $\bar{y}$;
- лингвистическая модель или база правил, представляющая множество *правил нечётких продукций*, с посылкой (A) и следствием (B) [5]:

$$R^{(k)} : A^k \rightarrow B^k, k = \overline{1, r}, \quad (7)$$

где r — число нечётких правил; k — номер правила; A^k и B^k — нечёткие множества такие, что:

$$A^k = A_1^k \times A_2^k \times \dots \times A_n^k \quad A_i^k \subseteq X_i, \quad (8)$$

$$B^k = B_1^k \times B_2^k \times \dots \times B_m^k \quad B_j^k \subseteq Y_j, \quad (9)$$

где X_i и Y_j — пространство входных и выходных переменных, при этом $i = \overline{1, n}$, $j = \overline{1, m}$;

- блок выработки решения (блок вывода), который формирует с использованием заданной базы правил на своём выходе одно или несколько нечётких множеств B , применив полученные на вход нечёткие множества A , при этом используется следующая последовательность шагов:

1. Агрегирование условий, составляющих посылку правила;
2. Активизация заключений, составляющих следствие правила;
3. Аккумуляция следствия — получение нечёткого множества (или множеств) представляющего следствие.

Рассмотрим реализацию СНВ для использования в качестве модуля расчёта кэш-рейтингов объектов в алгоритме кэширования, применяемом для web-прокси серверов. На вход СНВ подаются значения характеристик объекта системы, на выходе получаем кэш-рейтинг объекта. Используются известные характеристики объекта в трассе: дистанция последнего доступа, популярность за последнее время (частота доступов), размер объекта. Кроме того, в качестве входного

параметра впервые предлагается использовать характеристику *пространственной локальности web-ресурса*.

В традиционном смысле пространственная локальность применяется при кэшировании программных инструкций в оперативной памяти, когда близкими являются инструкции расположенные в одном блоке. Если общее *информационное пространство* web-среды рассматривать как единое множество адресов, то *близкими* с точки зрения пространственной локальности можно называть ресурсы, расположенные на одном web-сервере (либо в одной локальной сети, либо на одном прокси-сервере и т. д., в зависимости от решаемой задачи). При этом мера близости двух web-ресурсов в самом простом случае может быть установлена с использованием унифицированных идентификаторов этих ресурсов (англ. *Uniform Resource Identifier, URI*): например, все ресурсы, расположенные на одном доменном имени являются близкими. При этом далёкими по отношению к ним называются остальные ресурсы. Таким образом, мера близости двух web-ресурсов представляется функцией:

$$\mu(x_1, x_2) = \begin{cases} 0, & u_{x_1} \text{ далек от } u_{x_2}, \\ 1, & u_{x_1} \text{ близок к } u_{x_2}, \end{cases} \quad (10)$$

где x_1 и x_2 — численные идентификаторы web-ресурсов, как объектов системы; u_{x_1} и u_{x_2} — URI web-ресурсов с идентификаторами x_1 и x_2 .

Суть описываемой ситуации заключается в том, что обращение пользователя к некоторому информационному контенту web-сайта, как правило, влечёт за собой несколько параллельных асинхронных запросов к web-серверу на предоставление такой информации, как изображения, js-скрипты, таблицы стилей CSS и т. д. Таким образом, запрос данных с сервера подразумевает выполнение множественных запросов на получение объектов из множества *близких* ресурсов, а значит, кэш-рейтинги этих ресурсов должны быть повышены.

Для применения данного свойства в реальной системе кэширования требуется его характеризовать численным образом. Для этого введём в рассмотрение кортеж с LRU-организацией D_{uri} составленный из адресов доменов, к которым принадлежат ресурсы, запрашиваемые в трассе:

$$D_{uri} = (d_1, d_2 \dots d_n), \quad (11)$$

где d_i — доменное имя; i — позиция имени в кортеже.

Позиция имени в кортеже характеризует новизну обращений к ресурсам данного домена: на первой позиции расположено имя, к ресурсу которого было последнее обращение, т. е. позиция в кортеже D_{uri} представляет дистанцию последнего обращения к ресурсам соответствующего доменного имени.

Таким образом, на вход блока фаззификации поступают следующие значения:

- дистанция последнего обращения к объекту в трассе (OD — *object distance*), характеризующая свойство новизны — число позиций в трассе (либо число секунд) до предыдущего обращения к объекту, а если обращение к объекту первое в трассе, то $OD = -1$;
- популярность объекта (OP — *object popularity*) — число обращений к объекту;
- размер объекта (OS — *object size*) — размер объекта в заданных единицах информации (например, в байтах);
- характеристику пространственной локальности объекта (SL — *spatial locality*), которая представлена позицией домена web-ресурса в кортеже D_{uri} а если домен в кортеже ещё не представлен, то $SL = -1$.

Для каждого представленного значения выполняется операция фаззификации типа «сингтон» [5]. Полученные нечёткие множества подаются в блок вывода, на выходе которого регистрируется одно нечёткое множество, характеризующее *значимость* объекта для кэш-системы.

После дефаззификации данного нечёткого множества по методу центра тяжести, на выходе СНВ получаем кэш-рейтинг объекта.

Ключевым звеном описанной нечёткой системы, определяющим её функционирование, является лингвистическая модель, представленная множеством нечётких правил, которые в описанных ранее кэш-системах на базе СНВ определялись аналитически на базе известных участков трасс [6, 7]. Разработанный алгоритм реализует метод синтеза нечётких правил по результатам выполнения нечёткой кластеризации за счёт изменения функций принадлежности нечётких множеств, входящих в *посылку* правила [10]. Таким образом, представленный алгоритм кэширования на базе нечёткой логики адаптируется к изменяющимся характеристикам трассы.

Для получения входных данных процедуры кластеризации подаваемая на вход трасса объектов разбивается на участки. На каждом участке производится вычисление следующих характеристик объектов:

- оценка математического ожидания дистанции объекта (OD) на участке трассы;
- число обращений к объекту (OP) на участке;
- размер объекта (OS);
- оценка математического ожидания дистанции обращения к ресурсам домена (SL) на участке.

При этом учитываются только те объекты, число обращений к которым на участке больше некоторого заданного порогового значения T_{fuzzy} ($T_{fuzzy} > 1$). Полученные данные объединяются в векторы, подаваемые в качестве входных в процедуру кластеризации:

$$\bar{x}^t = (x_1, x_2, x_3, x_4, x_5), \quad (12)$$

где x_1 — оценка математического ожидания дистанции объекта на участке t ; x_2 — число обращений к объекту на участке t ; x_3 — размер объекта; x_4 — оценка математического ожидания дистанции обращения к ресурсам домена на участке t ; x_5 — число обращений на участке $t + 1$.

Для получения кластеров из множества полученных векторов используется алгоритм нечётких c -средних (англ. *Fuzzy C-Means*) [5]. При этом число кластеров устанавливается соответствующим числу желаемых правил нечётких продукций, синтез которых осуществляется по следующим правилам:

- x_1, x_2, x_3, x_4 — представляют посылку правила, x_5 — следствие правила;
- в качестве заключения правила выбирается центр соответствующего кластера;
- функции принадлежности нечётких множеств, входящих в посылку и следствие нечёткого правила, продуцируются путём проецирования степеней принадлежности соответствующего кластера на входные переменные [10].

Полученные таким образом нечёткие правила применяются в качестве базы СНВ. Число участков трассы, после которых выполняется адаптация, и размер участка являются параметрами представленного нечёткого алгоритма.

Результаты имитационного моделирования. Исследования проводились с применением имитационного моделирования на разработанном программном стенде [11]. Для проведения исследований был выбран метод событийно-регулируемого управления моделированием (англ. *trace-driven simulation*), а в качестве входных данных подавались трассы с прокси-сервера Донского государственного технического университета, использующего в качестве программного обеспечения популярный кэширующий сервер Squid. Трассы получены на основе журналов работы Squid в период: март-май, июль-август 2012 года. При этом для проведения экспериментов все трассы были приведены к одинаковому размеру и в них оставлены только те обращения к веб-ресурсам, которые могут кэшироваться прокси-сервером. Усреднённые статистические характеристики используемых реальных трасс представлены в таблице 1.

Таблица 1

Усреднённые параметры реальных трасс, используемых в исследованиях

Параметр	Значение
длина трассы	1.000.000
% числа объектов, запрошенных единожды	73,03 % $\pm$ 3 %
% трафика объектов, запрошенных единожды	65,70 % $\pm$ 3 %
средний размер объекта	18 Кб
максимальный размер объекта	68679 Кб
минимальный размер объекта	1 Кб
общий размер трафика	10,72 Гб

При проведении экспериментов размер кэш-памяти варьировался в пределах от 10 Мб до 300 Мб. Результаты проведённых экспериментов приведены ниже (рис. 2, табл. 2).

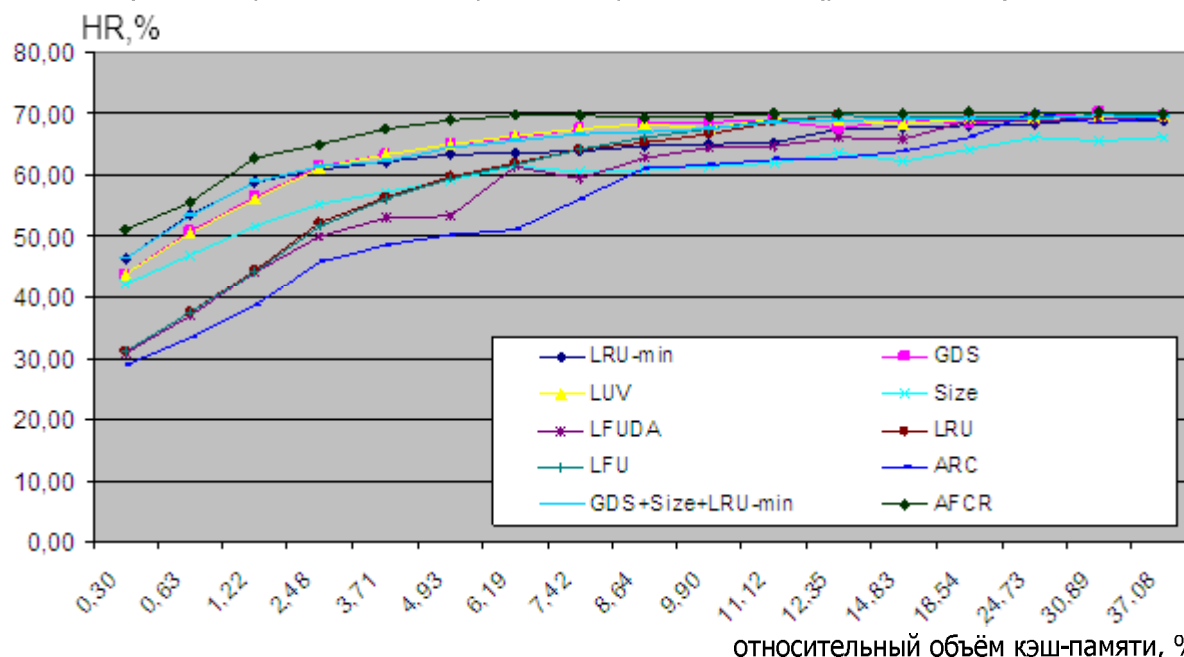


Рис. 2. Динамика изменения эффективности стратегий замещения на реальных трассах для объектов, запрошенных более одного раза

Таблица 2

Сравнение стратегий замещения на реальных трассах для объектов запрошенных более одного раза

	LFU	LRU	Size	LFUDA	LRU-min	GDS	LUV	ARC	AF CR
HR	60,350	60,347	59,201	58,209	62,288	63,492	63,355	58,236	68,031
BHR	25,139	26,363	26,845	22,358	25,213	24,101	25,213	24,842	27,809

Выводы. Разработанная специально для прокси-серверов адаптивная нечёткая стратегия замещения AF CR на имитационном стенде показала превышение значений рейтинга кэш-попаданий, как основного критерия эффективности алгоритмов кэширования, известных стратегий замещения в среднем на 8 % и с вероятностью 0,95 не менее, чем на 5 %.

Библиографический список

1. Arlitt, M. F. Performance Evaluation of Web Proxy Cache Replacement Policies / Martin Arlitt, Rich Friedrich, Tai Jin // Internet Systems and Applications Laboratory. — October, 1999.

2. Danzig, P. B. A Case for Caching File Objects Inside Internetworks / Peter B. Danzig, Richard S. Hall, Michael F. Schwartz // ACM SIGCOMM Computer Communication Review. — Volume 23. — Issue 4. — 1993.
3. Yang, Q. Web-Log Mining for Predictive Web Caching. / Q. Yang, and H. H. Zhang // IEEE Transactions on Knowledge and Data Engineering. — 2003. — Volume 15. — Number 4.
4. Жуков, А. И. Использование информационных систем и технологий в целях удовлетворения информационных потребностей / А. И. Жуков, А. Г. Сорокин. — Красноярск : Научно-инновационный центр, 2012. — 68 с.
5. Рутковский, Л. Методы и технологии искусственного интеллекта / Л. Рутковский; [перевод с польского И. Д. Рудинского]. — Москва : Горячая линия — Телеком, 2010. — 520 с.
6. Calzarossa, M. C. A Fuzzy Algorithm for Web Caching [Электронный ресурс] / Maria Carla Calzarossa, Giacomo Valli. — Режим доступа: <http://peg.unipv.it/publications/PDF/Proxy.pdf> (дата обращения: 12.11.2012).
7. Sabeghi, M. Using Fuzzy Logic to Improve Cache Replacement Decisions / Mojtaba Sabeghi, and Mohammad Hossein Yaghmaee // IJCSNS International Journal of Computer Science and Network Security. — 2006. — Volume 6. — No. 3.
8. Жуков, А. И. Модель адаптивного векторного управления стохастическим гибридным алгоритмом кэширования / А. И. Жуков // Вестник Донского гос. техн. ун-та. — 2012. — № 5. — С. 19–29.
9. Лю, Б. Теория и практика неопределённого программирования / Б. Лю. — Москва : БИНОМ. Лаборатория знаний, 2005. — 416 с.
10. Штовба, С. Д. Введение в теорию нечётких множеств и нечёткую логику / С. Д. Штовба. — Винница : Континент-Прим, 2003. — 198 с.
11. Жуков, А. И. Программный стенд для исследования эффективности алгоритмов кэширования / А. И. Жуков // Системный анализ, управление и обработка информации : Труды 1-го Международного семинара студентов, аспирантов и учёных. — Ростов-на-Дону : ИЦ ДГТУ, 2010. — С. 249–253.

Материал поступил в редакцию 14.11.2012.

References

1. Arlitt, M.F., Friedrich, R., Tai Jin. *Performance Evaluation of Web Proxy Cache Replacement Policies*. Internet Systems and Applications Laboratory, October, 1999.
2. Danzig, P.B., Hall, R.S., Schwartz, M.F. *A Case for Caching File Objects Inside Internetworks*. ACM SIGCOMM Computer Communication Review, 1993, vol. 23, iss. 4.
3. Yang, Q., Zhang, H.H. *Web-Log Mining for Predictive Web Caching*. IEEE Transactions on Knowledge and Data Engineering, 2003, vol. 15, no. 4.
4. Zhukov, A.I., Sorokin, A.G. *Ispolzovaniye informatsionnykh sistem i tekhnologiy v tselyakh udovletvoreniya informatsionnykh potrebnostey*. [Data systems and technologies application for meeting information needs.] Krasnoyarsk: Nauchno-innovatsionnyy tsentr, 2012, 68 p. (in Russian).
5. Rutkovsky, L. [Translation from Polish by Rudinsky, I.D.] *Metody i tekhnologii iskusstvennogo intelekta*. [Methods and technologies of artificial intellect.] Moscow: Goryachaya liniya – Telekom, 2010, 520 p. (in Russian).
6. Calzarossa, M.C., Valli, G. *A Fuzzy Algorithm for Web Caching*. Available at: <http://peg.unipv.it/publications/PDF/Proxy.pdf>.
7. Sabeghi, M., Yaghmaee, M.H. *Using Fuzzy Logic to Improve Cache Replacement Decisions*. IJCSNS International Journal of Computer Science and Network Security, 2006, vol. 6, no. 3.

8. Zhukov, A.I. *Model adaptivnogo vektornogo upravleniya stokhasticheskim gibridnym algoritmom keshirovaniya*. [Model of adaptive vector control of stochastic hybrid caching algorithm.] *Vestnik of Don State Tech. University*, 2012, no. 5, pp. 19–29 (in Russian).

9. Liu, B. *Teoriya i praktika neopredelennogo programmirovaniya*. [Theory and Practice of Uncertain Programming.] Moscow: BINOM. Laboratoriya znaniy, 2005, 416 p. (in Russian).

10. Shtovba, S.D. *Vvedeniye v teotiyu nechetkikh mnozhestv i nechetkuyu logiku*. [Introduction to fuzzy-set theory and fuzzy logics.] Vinnytsia: Kontinent-Prim, 2003, 198 p. (in Russian).

11. Zhukov, A.I. *Programmnyy stend dlya issledovaniya effektivnosti algoritmov keshirovaniya*. [Programmed testbed for caching algorithm efficiency analysis.] *Sistemnyy analiz, upravleniye i obrabotka informatsii: Trudy 1-go Mezhdunarodnogo seminara studentov, aspirantov i uchenykh*. [System analysis, administration, and data processing: Proc. I Int. Seminar for students, postgraduates and scholars.] Rostov-on-Don: DSTU Publ. Centre, 2010, pp. 249–253 (in Russian).

ADAPTIVE FUZZY CACHING ALGORITHM FOR PROXY SERVERS

A. I. Zhukov

(Don State Technical University)

The problem of increasing the efficiency of cache systems using artificial intelligence techniques is considered. The formalized statement of raising the cache effectiveness on the basis of the set-theoretic mathematical tools is described. The fuzzy logic application technology for determining the victim which implements the calculation of cache values for object in cache memory is considered. It is first offered to use spatial locality of the web-resource as one of its basic characteristics applying to the input of the fuzzy logic system. The definition of this characteristic is based on the uniform resource identifier URI. The technique for the adaptation of fuzzy systems through the synthesis of the linguistic model using the method of fuzzy clustering is defined. Besides, the experimental study of the algorithm developed on the simulation stand for traces obtained through the proxy cache Squid is resulted.

Keywords: cache system, fuzzy logic, fuzzy clustering, adaptive caching system.