Розділ 2. Інформаційні технології та телекомунікації

УДК 631

С.С. Батыр, А.В. Хорхордин
Донецкий национальный технический университет, г. Донецк,
Кафедра «Автоматики и телекоммуникаций»
E-mail: sbatyr@gmail.com

ОБОСНОВАНИЕ БИЛИНЕЙНОСТИ АЛГОРИТМОВ УПРАВЛЕНИЯ ПЕРЕГРУЗКОЙ ПРОТОКОЛА TCP

Аннотация
Батыр С.С., Хорхордин А.В. Обоснование билинейности алгоритмов управления перегрузкой протокола TCP. *Проведен обзор и анализ существующих алгоритмов управления перегрузкой для протокола гарантированной доставки данных TCP. Передатчик, работающий по протоколу TCP, рассмотрен как объект в теории автоматического управления. Выделены входные и выходные переменные, построена его структурная схема. Для ряда алгоритмов проведен синтез структурных схем регуляторов и соответствующих уравнений. Согласно проведенному анализу использующиеся регуляторы являются билинейными.*
Ключевые слова: *перегрузка, билинейная система, алгоритм управления перегрузкой, TCP Reno, TCP Vegas.*

Введение. Современные средства коммуникаций используют преимущественно каналы связи с пакетной коммутацией. Это позволяет повысить эффективность использования каналов, однако приводит к снижению надежности доставки информации. При перегрузке канала с пакетной коммутацией возможна ситуация, когда пакет на входе канала не поместится во входной буфер, а будет отброшен.

Для обеспечения гарантированной доставки по каналам без гарантированной доставки были разработаны специальные протоколы. Один из наиболее популярных – это протокол Transmission Control Protocol (TCP) из стека протоколов TCP/IP. Он проектировался для доставки данных служб обмена файлами, электронной почты и т.п. Для этих служб основным требованием была корректная доставка информации. При этом время доставки не было критичным фактором. Развитие мультимедийных технологий поставило задачу доставки сообщений при соблюдении высоких требований качества обслуживания: малая задержка и малая вариация задержки. Эта задача решается при помощи регулирования скорости передачи и приема сообщений.

Управление скоростью передачи осуществляется с помощью алгоритма управления перегрузкой (TCP congestion control algorithm). Именно он управляет шириной скользящего окна и контролирует интенсивность передачи данных TCP-передатчиком. В нем реализованы обработчики следующих событий: прием ответа-подтверждения доставки от приемника, потеря пакета в буфере маршрутизатора (на пути доставки пакета), превышение времени ожидания ответа приемника.

Изначально протокол TCP проектировался для распределенных сетей со значительным временем доставки. Долгая жизнь данного протокола, как и всего стека TCP/IP v4, не предполагалась. Поэтому изначально в протокол был заложен простейший алгоритм с минимальной вычислительной нагрузкой на процессор системы. Производительность процессоров за последние 30 лет выросла многократно, пропускная

способность каналов связи увеличилась в тысячи раз, а время доставки для высокоскоростных локальных и глобальных сетей составляет доли микросекунд. В соответствии с техническим прогрессом развивался и протокол TCP.

Отрицательное влияние простейшего алгоритма управления перегрузкой на эффективную пропускную способность соединения было отмечено еще в конце 80-ых – начале 90-ых годов XX в. Тогда же начали предлагать новые, более совершенные версии, хотя и в них присутствуют некоторые недостатки. Были предложены алгоритмы, позволяющие учитывать состояние канала связи (TCP Vegas), учитывающие высокий уровень потерь в каналах связи (TCP Westwood), высокие пропускные способности локальных сетей новых поколений (TCP Hamilton).

Быстрый рост беспроводных средств коммуникаций (спутниковые каналы, сети мобильной связи, WiFi и др.) привели к вынужденной адаптации протокола TCP к их особенностям: большая пропускная способность канала при значительной задержке доставки. Были созданы специальные версии алгоритмов управления перегрузкой, учитывающие эти особенности. К таким алгоритмам относятся TCP BIC и TCP CUBIC(менее агрессивная в плане занятия канала связи версия TCP BIC).

Однако, несмотря на все усилия разработчиков, эффективное предупреждение перегрузки и значительное снижение объема переотправляемых данных не было достигнуто. Взрывной рост числа абонентов сети Интернет, увеличение количества служб с самыми разнообразными требованиями к качеству обслуживания привело к тому, что определение состояния каналов связи на маршруте доставки стало затруднительным. Это практически сводит на нет эффективность работы существующих алгоритмов управления перегрузкой.

Поэтому продолжается дальнейшая разработка и совершенствование алгоритмов управления. Последние годы значительное внимание уделяется синтезу алгоритмов управления перегрузкой и управления очередями маршрутизатора с применением теории управления. При этом в качестве основной модели процесса передачи данных используется потоковая модель, предложенная в рамках исследования [1]. Разработки, базирующиеся на данной модели, освещены в работах: [2], [4].

При построении данной модели принят ряд допущений: передаваемые данные представлены непрерывными потоками, все потоки рассматриваемого участка сети принимаются синхронизированными, нелинейная модель передатчика TCP линеаризирована в области рабочей точки, задающей размер очереди, ширину скользящего окна и уровень потерь на выходной очереди маршрутизатора. К недостаткам применения линеаризированной модели можно отнести такие допущения как фиксированное количество источников, невозможность учета изменения маршрута доставки сообщений и т.п. Данные допущения могут оказать значительное влияние на результаты при моделировании распределенных протоколов обмена данными (BitTorrent, DC, Emule) и моделировании сетей беспроводного доступа со значительными колебаниями времени доставки пакета.

Цель исследования. Таким образом, необходимо построение модели передатчика протокола TCP, который позволил бы учесть возможное резкое изменение времени доставки и изменение количества передатчиков в процессе работы сети.

Описание объекта. На сегодняшний день разработано значительное количество алгоритмов управления перегрузкой. Для проведения анализа и построения модели составим перечень основных алгоритмов с учетом популярности их использования в современных средствах коммуникации. Ограничим круг рассматриваемых устройств персональными компьютерами, серверами и рабочими станциями. Устройства класс мобильные телефоны, смартфоны и коммуникаторы рассматриваться не будут в виду специфики их назначения и малодоступности информации о их внутреннем устройстве.

Функции управления процессом передачи данных возложены на операционные системы персональных компьютеров и серверов. В наиболее распространенных операционных системах используются следующие алгоритмы:

- Windows NT kernel based (Windows NT, 2000, XP, Vista, 7) – используется алгоритм Compound TCP; данный алгоритм разработан на базе TCP Reno/NewReno + дополнительные модификации, проведенные разработчиками Microsoft; оптимизирован для сетей с большой пропускной способностью и варьирующейся в широких пределах задержкой доставки.

- Linux kernel based (Ubuntu, Debian и др.) – по умолчанию используется TCP Vegas (задается при компиляции ядра), возможна смена алгоритма во время работы системы без перезагрузки ОС;

- FreeBSD kernel based (FreeBSD, MacOSX, iOS) – по умолчанию используется TCP Reno, возможна смена на любой из существующих.

Исходя из представленного и дополнительных источников [5], получим следующий список наиболее часто используемых алгоритмов управления перегрузкой. Представим алгоритмы вместе с их краткой характеристикой.

TCP Tahoe – первая реализация алгоритма управления перегрузкой. Представляет собой импульсный дискретный регулятор, изменяющий свое состояние при получении пакета подтверждения доставки либо истекания таймера ожидания доставки. На сегодняшний день применяется лишь во встраиваемых системах с малой производительностью. Используется в качестве базового алгоритма при сравнении эффективности работы алгоритмов.

TCP Reno/NewReno – усовершенствованный вариант TCP Tahoe. Введен дополнительный алгоритм «быстрого восстановления». Один из основных применяющихся алгоритмов на сегодняшний день. Модификация NewReno позволяет передатчику переходить к «быстрому восстановлению» без ожидания истечения таймера.

TCP Vegas – алгоритм с учетом состояния канала. Данная реализация, с одной стороны, обеспечивает достаточно высокую пропускную способность соединения TCP, а с другой — низкую, относительно базового TCP Reno, вероятность возникновения перегрузки. Столь сложная задача решается совместно с использованием двух следующих подходов: 1) заблаговременное обнаружение перегрузки или состояния близкого к ней в рамках транзитного узла (маршрутизатора) и 2) предсказание потери сегмента на базе статистики изменения значения время оборота кадра между передатчиком и приемником RTT — чем выше это значение, тем выше нагрузка на маршруте до получателя. К недостаткам можно отнести: 1) уровень пропускной способности сети определяется при установлении соединения; 2) уровень не изменяется до окончания соединения, соответственно, передатчик слабо реагирует на освобождение сетевых ресурсов.

TCP CUBIC – алгоритм предназначен для сетей с большой пропускной способностью и значительной задержкой – спутниковые каналы передачи данных, сети мобильной связи. Имеет нелинейную статическую характеристику, адаптированную под условия работы. Менее агрессивен, чем алгоритм TCP BIC. Мало применим в локальных сетях.

Проведем анализ передатчика TCP с применением методов пространства состояний. Это позволит получить модель TCP-передатчика, которую будет возможно использовать в исследованиях процессов передачи данных.

Основная часть. На начальном этапе анализа воспользуемся методом «черный ящик». Это позволит определить входные и выходные переменные. После этого перейдем к «серому ящику» и определим переменные состояния и взаимодействие между ними.

Для работы передатчика необходимы данные к передаче. Примем ряд допущений, которые упростят последующий анализ: 1) количество данных к передаче неограниченно; 2) интенсивность поступления данных превышает пропускную способность каналов связи и может не учитываться при построении модели сети; 3) характер содержимого передаваемых

данных не учитывается. В терминах ТАУ с принятием допущений данные к передаче могут быть представлены как ступенчатое входное воздействие, обозначим как d.

Выходом передатчика примем интенсивность передачи данных χ_{TCP}. Преимуществом данной величины является возможность суперпозиции интенсивности потоков данных от нескольких источников.

Алгоритмы управления перегрузкой при своей работе используют два сигнала: 1) подтверждение о доставке u_{RTT} от получателя данных – сигнал свидетельствует о корректной доставки данных и разрешает дальнейшую передачу; 2) отбрасывание пакета в маршрутизаторе u_{lost} – сигнал свидетельствует о перегрузке очереди на маршруте доставки пакета. В терминах ТАУ данные сигналы можно представить как единичные импульсы в цепях обратной связи.

Таким образом, получим следующую модель типа «черный ящик»(см. рис.1). Модель представляет собой систему типа MISO, где один управляемы вход и два входа обратной связи воздействуют на один выход.

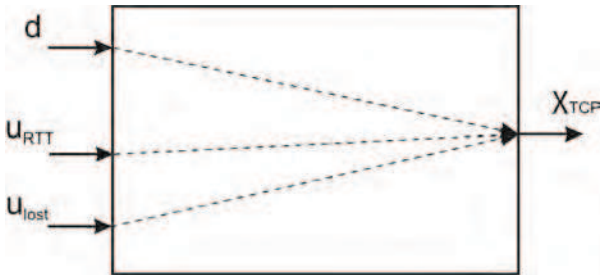

Рисунок 1 – Модель черного ящика

Процесс работы передатчика TCP подробно описан в [6]. Методика работы алгоритмов управления перегрузкой описана в [3]. На основе этих материалов построена структурная схема передатчика TCP (см. рис.3).

Основной переменной состояния передатчика будет длина окна передачи W, определяющая объем данных отправляемых до получения следующего подтверждения. Её величина определяется алгоритмом управления перегрузкой и она непосредственно влияет на выход системы – интенсивность передачи.

Как указывалось выше, выходной величиной принята интенсивность передачи данных χ_{TCP}. Данная величина, согласно описанию [3], будет функцией от длины окна передачи W и времени оборота кадра RTT. Возможны два представления. Первое предлагает амплитудную модуляцию сигнала интенсивности, но при этом величина интенсивности будет определяться как:

$$\chi_{TCP} = \frac{W}{RTT} \tag{1}$$

Этот вариант используется при потоковом моделировании сетей, но представляет значительные трудности при анализе системы как дискретной, так как значение RTT возможно определить лишь после получения подтверждения о доставке.

Второй вариант представления: как импульсы единичной амплитуды и переменной длительности. Интенсивность отправки данных через сетевой интерфейс $h_{network}$=[пакет/с] заранее известна и определяется конструктивными особенностями оборудования. Получение сигнала подтверждения будет указывать на момент начала отправки следующей порции данных. Таким образом, блок ширина окна – интенсивность передачи будет представлять собой широтно-импульсный модулятор. Уравнения, описывающие работу ШИМ, представлены в (1), графики, поясняющие работу системы – на рис.2.

$$\begin{cases} t_{0-1} = t_{u_{RTT}} \\ t_{1-0} = t_{u_{RTT}} + \dfrac{W}{h_{network}} \end{cases} \quad (2)$$

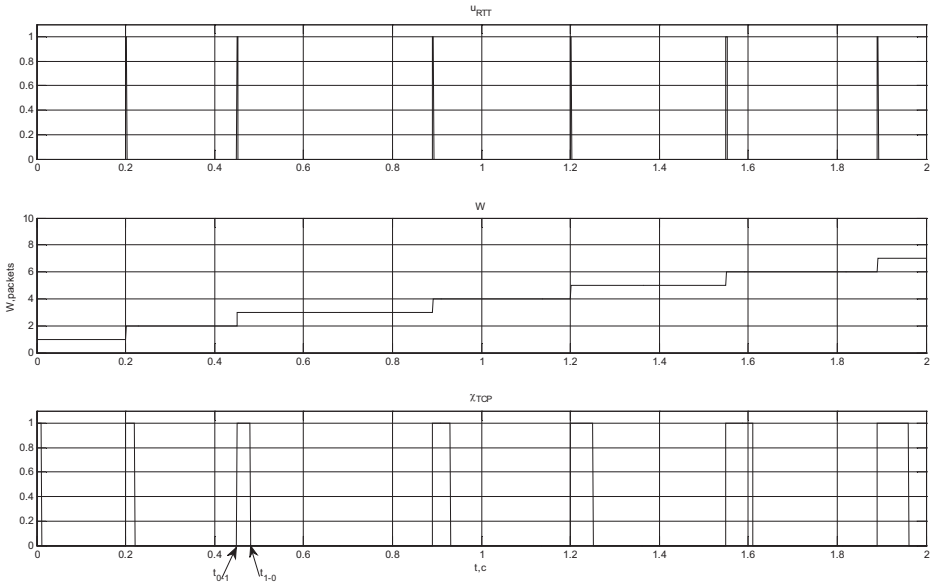

Рисунок 2 – Иллюстрация, поясняющая работу преобразователя длина окна – интенсивность передачи

При применении широтно-импульсной модуляции информационная часть сигнала не зависит от времени оборота кадра, при этом остается возможным суммирование интенсивностей нескольких источников. При построении модели передатчика применим второй вариант. В терминах ТАУ это будет нелинейное звено широтно-импульсного модулятора.

Ширина окна передачи задается величиной выхода регулятора алгоритма управления перегрузкой K_w и зависит от входной переменой d. Таким образом, получим уравнение:

$$W = K_w \cdot d \quad (3)$$

В выражении (3) переменная состояния системы K_W умножается на входное воздействие – системы такого типа относят к билинейным.

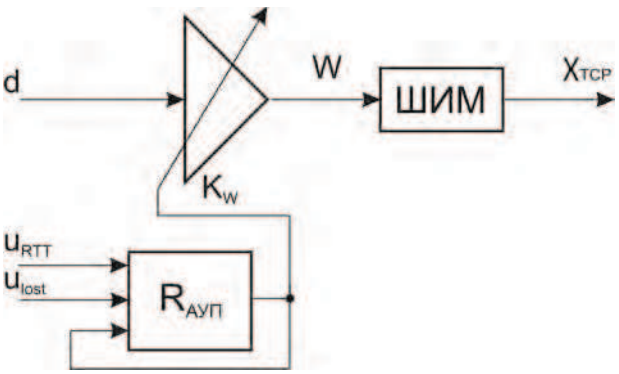

Рисунок 3 – Структурная схема передатчика TCP

Рассмотрим алгоритм управления перегрузкой TCP Reno. При нормальной ситуации длина окна передачи меняется циклически. Длина окна увеличивается до тех пор, пока не произойдет потеря сегмента. TCP-Reno имеет две фазы изменения размера окна: фаза медленного старта и фаза избежания перегрузки. При получении отправителем подтверждения доставки (u_{RTT}=1) в момент времени t + t_{RTT} [сек], текущее значение размера окна перегрузки $K_w(t)$ преобразуется в $K_W(t + t_{RTT})$ согласно (4):

$$K_W\left(t+t_{RTT}\right)=\begin{cases} \textit{фаза медленного старта} \\ K_W\left(t\right)+1,\ \textit{если}\ K_W\left(t\right)<ssth\left(t\right) \\ \textit{фаза исключения перегрузки} \\ K_W\left(t\right)+\dfrac{1}{K_W\left(t\right)},\ \textit{если}\ K_W\left(t\right)\geq ssth\left(t\right) \end{cases} \tag{4}$$

где ssth(t) [пакетов] - значение порога, при котором TCP переходит из фазы медленного старта в фазу исключения перегрузки. Когда TCP детектирует потерю пакета согласно алгоритму быстрой повторной передачи (u_{lost}=1), cwnd(t) и ssth(t) обновляются следующим образом:

$$ssth\left(t\right)=\frac{K_W}{2};\ K_W=ssth(t) \tag{5}$$

TCP Reno после этого переходит в фазу быстрого восстановления. В этой фазе размер окна увеличивается на один пакет, когда получается дублированное подтверждение. С другой стороны, $K_W(t)$ делается равным ssth(t), когда приходит не дублированный отклик для пакета, посланного повторно.

На основе выражений (4) и (5) можно составить общее уравнение регулятора АУП для TCP Reno. Исключим из рассмотрения фазы медленного старта и исключения перегрузки, которые можно отнести к переходным процессам, которые ведут к фазе быстрого восстановления. Также примем во внимание, что сигналы u_{RTT} и u_{lost} прийти одновременно не могут. Получим следующее выражение:

$$K_W\left(t+t_{RTT}\right)=\left(K_W+1\right)u_{RTT}-\frac{1}{2}K_W u_{lost} \tag{6}$$

Полученное выражение дискретно по времени. Переменная состояния K_W умножается на входное воздействие u_{RTT} и u_{lost} – системы такого типа относят к билинейным.

Рассмотрим алгоритм управления перегрузкой TCP Vegas. Алгоритм состоит из следующих этапов:

1) при установлении соединения определяется величина MTU и измеряется минимальное время оборота кадра между передатчиком и приемником RTT_{min}. Данная величина принимается как задержка доставки для сети без полезной нагрузки.

2) длина окна передачи W устанавливается на уровне 1 пакет и начинается работа алгоритма. При получении подтверждения о доставке u_{RTT} производится расчет корректирующего воздействия на размер окна dW.

Согласно значению таймера RTT устанавливается значение текущего времени оборота кадра RTT_{cur} и производится оценка отличия Diff идеальной пропускной способности канала от реальной (см. выражение (7)).

$$Diff=\frac{W}{RTT_{\min}}-\frac{W}{RTT} \tag{7}$$

И производится вычисление изменение длины окна передачи:

$$\begin{cases} if\ Diff < \alpha / RTT_{\min} \rightarrow W := W + 1 \\ if\ Diff > \beta / RTT_{\min} \rightarrow W := W - 1 \end{cases} \quad (8)$$

Константы α и β – это параметры алгоритма, их значения принимаются из заданного диапазона: $\alpha \in [1:3]$, $\beta \in [2:4]$, причем $\alpha < \beta$.

Описанный алгоритм можно представить в виде следующей структурной схемы:

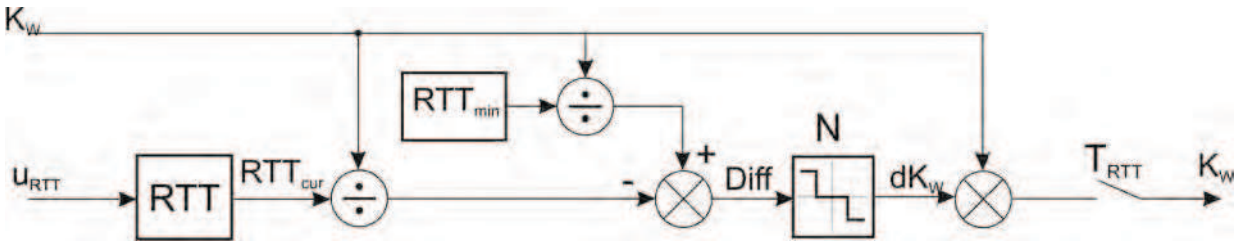

Рисунок 4 – Структурная схема алгоритма управления перегрузкой TCP Vegas

Система (8) представлена в виде трехпозиционного реле, которое определяет изменение окна передачи из возможного множества значений (-1; 0; 1). На выход блока поступает новое значение длины окна передачи K_W.

Представленную схему можно описать следующим выражением:

$$K_W\left(t + t_{RTT}\right) = K_W\left(t\right) + N\left(\frac{K_W\left(t\right)}{RTT_{\min}} - \frac{K_W\left(t\right)}{RTT_{cur}\left(t + t_{RTT}\right)}\right) \quad (9)$$

$$RTT_{cur}\left(t + t_{RTT}\right) = t_{RTT}$$

Данное выражение описывает нелинейную систему с трехпозиционным реле и является дискретным по времени.

Рассмотрев структурные схемы передатчика TCP и регуляторов АУП разных видов получим, что описанные системы относятся к нелинейным системам управления. Их можно отнести к подвиду нелинейных систем – билинейным. В общем виде билинейная система описывается уравнением вида:

$$\dot{\vec{x}} = A \cdot \vec{x} + B \cdot u + K \cdot \vec{x} \cdot u \quad (10)$$

В подобных системах происходит умножение переменных состояния на входные воздействия. Анализ подобных систем выходит за рамки классической теории автоматического управления, однако возможно использование современной теории управления (метод переключения состояний) и специальной теории управления билинейными системами.

Выводы. Проведенный анализ алгоритмов управления перегрузкой показал, что они представляют собой нелинейные дискретные регуляторы, задающие размер окна передачи в зависимости от сигналов обратной связи по сообщениям: подтверждение получения пакета адресатом и потеря пакета в маршрутизаторе на маршруте следования. Регуляторы, построенные на основе алгоритмов TCP Reno и TCP Vegas, представляют собой билинейные

дискретные системы. Их анализ необходимо проводить с применением специальной теории управления для билинейных систем.

Литература

1. Benyuan Liu, Yang Guo, Jim Kurose, Don Towsley,Weibo Gong Fluid Simulation of Large Scale Networks: Issues and Tradeoffs - In Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications – University of Massachusetts Amherst, MA, U.S.A. – 1999, p.2136--2142

2. Van Jacobson, Michael J. Karels. Congestion Avoidance and Control(1988). *Proceedings of the Sigcomm '88 Symposium*, vol.18(4): pp.314–329. Stanford, CA. August, 1988. This paper originated many of the congestion avoidance algorithms used in TCP/IP.

3. RFC 2581- TCP Congestion Control – Адрес статьи: http://www.faqs.org/rfcs/rfc2581.html – Время доступа: 02.02.2011

4. Thomas Bonald "Comparison of TCP Reno and TCP Vegas via Fluid Approximation" Rapport de recherché n 3563, Unite de recherché INRIA Sophia Antipolis Cedex (France) – 1998 – 34 pages, Электронный доступ: http://citeseerx.ist.psu.edu/viewdoc/summary?doi= 10.1.1.10.2014 Время доступа: февраль 2011 г.

5. TCP congestion avoidance algorithm (Wikipedia) – Адрес статьи: http://en.wikipedia.org/wiki/TCP_congestion_avoidance_algorithm – Время доступа: 02.02.2011

6. RFC 793 - Transmission Control Protocol – Адрес статьи: http://www.faqs.org/rfcs/rfc793.html – Время доступа: 02.02.2011.

Надійшла до редакції:
01.03.2011

Рекомендовано до друку:
д-р техн. наук, проф. Чичикало Н.І.

Abstract

Batyr S.S., Khorkhordin A.V. Justification of bilinearity of algorithms TCP congestion control. *A review and analysis of existing algorithms for TCP congestion control. The TCP transmitter considered as an object of automatic control theory. Input and output variables were isolated, a block diagram was built. For a number of algorithms were synthesized structural schemes of the controllers and relevant equations. According to the analysis used controllers are bilinear.*
Keywords: *congestion, bilinear system, congestion control system, перегрузка, TCP Reno, TCP Vegas.*

Анотація

Батир С.С., Хорхордин О.В. Обґрунтування білінійності алгоритмів управління перевантаженням протоколу TCP. *Проведено огляд й аналіз існуючих алгоритмів управління перевантаженням для протоколу гарантованої доставки даних TCP. Передатчик, що працює за протоколом TCP, розглянуто як об'єкт в теорії автоматичного управління. Визначено вхідні та вихідні змінні, побудована його структурна схема. Для ряду алгоритмів проведено синтез структурних схем регуляторів й відповідних рівнянь. Згідно з проведеним аналізом існуючі регулятори є білінійними.*
Ключові слова: *перевантаження, білінійна система, алгоритм керування перевантаженням, TCP Reno, TCP Vegas.*