Повышение эффективности повторной передачи ошибочных пакетов данных по компьютерной сети

Данные передаются по компьютерным сетям в виде пакетов. Часть из них может теряться, например, из-за помех в линиях связи или из-за переполнения блоков буферной памяти устройств, расположенных по маршруту, пролегающему между источником и приёмником данных. В зависимости от протокола обмена потери пакетов либо остаются невосполнимыми (обычно это происходит при передаче речи в реальном времени), либо осуществляется их повторная пересылка. Последняя вызывает нежелательные задержки передачи данных. В этой статье приведено сравнение наиболее простых способов повторной передачи пакетов и рассмотрен новый, более эффективный способ [1].

Традиционные способы повторной передачи ошибочных пакетов по компьютерной сети

Прежде чем показать преимущества предлагаемого усовершенствования [1], рассмотрим наиболее простые способы повторной передачи ошибочных (потерянных) пакетов.

Один из них (Stop and Wait ARQ) основан на индивидуальном подтверждении правильности прохождения каждого пакета данных из узла А в узел В компьютерной сети (рис. 1) через объединённые линиями связи промежуточные узлы, в данном примере, через маршрутизаторы М1 — М3.

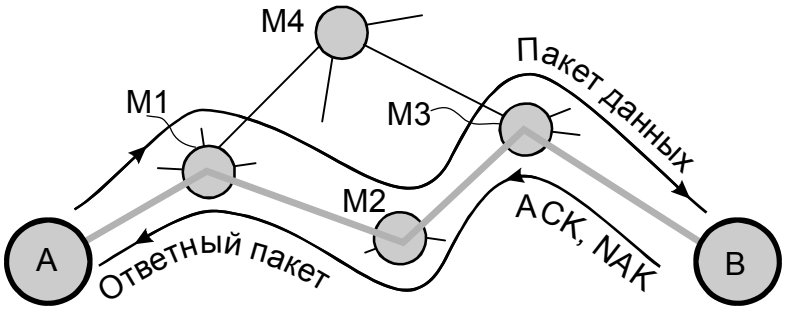

Рис. 1. Система передачи данных с подтверждением приёма каждого пакета

В установившемся режиме узел А посылает узлу В очередной пакет данных и ждёт подтверждения или неподтверждения правильности его получения. Узел В принимает этот пакет, проверяет его контрольную сумму CRC (Cyclical Redundancy Check) и, если ошибок не обнаружено, посылает узлу А ответный пакет ACK (Acknowledgment). Если узел В обнаружил ошибку в принятом пакете, то он посылает узлу А ответный пакет NAK (No Acknowledgment), который рассматривается узлом А как запрос повторной передачи пакета. Этот запрос немедленно удовлетворяется.

Если по истечении максимального времени ожидания узел А не получает ни того, ни другого ответа (ACK или NAK), то он повторяет выдачу потерянного или чрезмерно задержавшегося в пути пакета данных. В заголовке каждого пакета данных содержится его номер (например, принадлежащий циклическому диапазону 0 — 31), поэтому узел В ведёт учёт их поступления.

Описанный способ прост в реализации, объём выходной буферной памяти узла А минимален и составляет один пакет. В этой памяти хранится копия ранее отправленного пакета. При получении положительного ответа (ACK) от узла В копия уничтожается и буферная память принимает новый пакет одновременно с его выдачей в маршрутизатор М1.

Способ применим к системам с высоким уровнем помех в каналах связи. Оптимальная тактика передачи данных в таких условиях — “медленное, но верное” продвижение пакетов из узла А в узел В.

Недостаток данного способа — низкая скорость передачи данных из-за вынужденных прерываний их потока в связи с ожиданием получения ответа (АСК) на каждый посланный пакет.

Этот недостаток частично устранён в системах, использующих групповую передачу пакетов. При обнаружении ошибки хотя бы в одном из пакетов осуществляется повторная передача всей группы из n пакетов (Go Back n ARQ). Идея этого способа поясняется рис. 2.

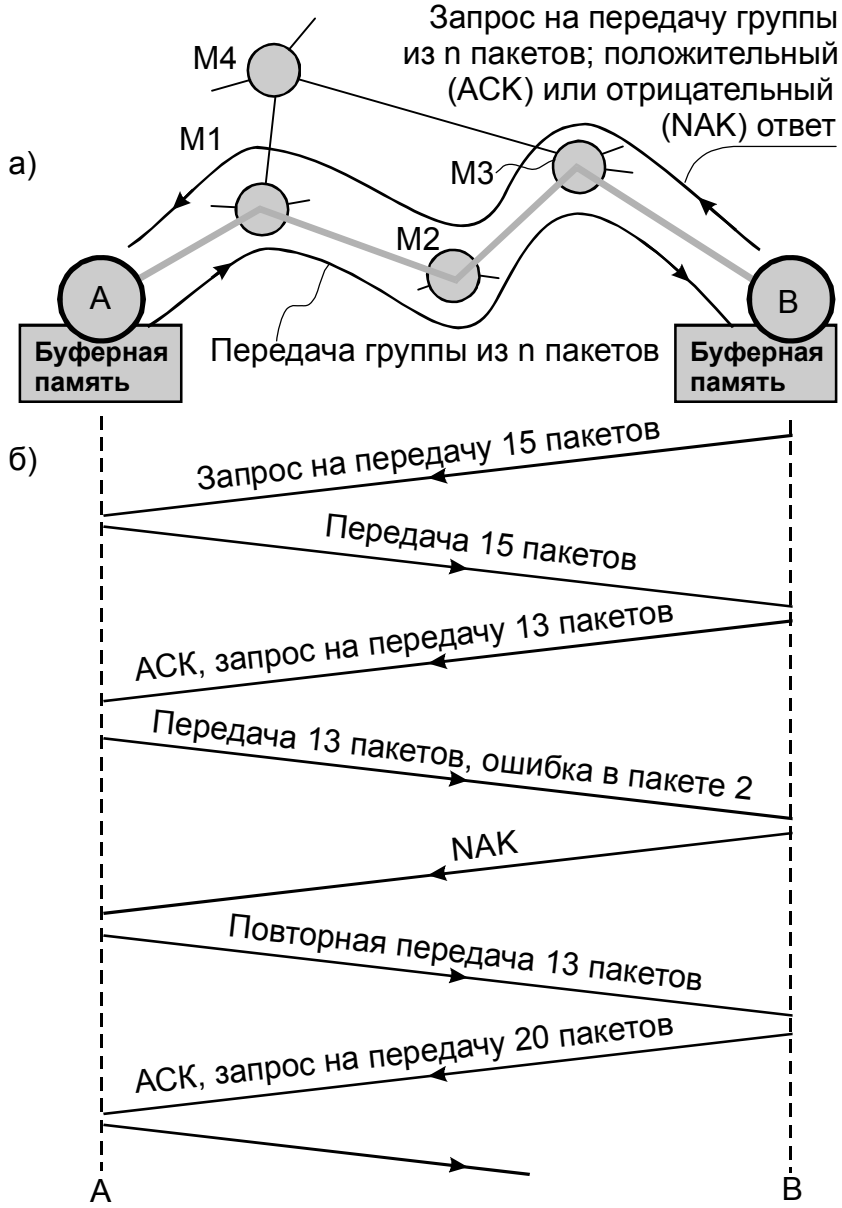

Рис. 2. Групповая передача пакетов: *а* — фрагмент сети; *б* — последовательность событий передачи

Рассмотрим один из возможных сценариев взаимодействия источника данных — узла А с приёмником — узлом В. Узлы А и В содержат блоки буферной памяти. Объём выходной буферной памяти узла А должен быть не меньшим объёма входной буферной памяти узла В.

После установления связи между узлами А и В последний оценивает степень заполнения своей входной буферной памяти и, убедившись в наличии достаточного свободного места, посылает узлу А запрос на передачу n = 15 пакетов (рис. 2, *б*). В ответ на этот запрос узел А посылает узлу В блок из пятнадцати пакетов, следующих друг за другом без пауз. Узел В проверяет каждый пакет блока и размещает их во входной буферной памяти.

Так как в данном случае все пакеты блока безошибочны, узел B посылает узлу A пакет ACK подтверждения приёма блока и (в этом же пакете) новый запрос на передачу тринадцати пакетов. Узел A уничтожает в выходной буферной памяти успешно переданную группу из пятнадцати пакетов и на освободившееся место помещает новую группу из тринадцати пакетов. Эта группа единым блоком пересылается в узел B.

Но на этот раз, как предполагаем, один из пакетов (с номером 2) прибывает на вход узла B с некорректируемой ошибкой. Узел B обнаруживает ошибочный пакет, прекращает приём, дожидается окончания передачи группы и посылает узлу A пакет NAK — запрос повторной передачи. Узел A вновь посылает в узел B группу из тех же тринадцати пакетов, которые только что были выданы. В данном сценарии повторная передача завершена успешно, узел B уведомляет об этом партнёра и одновременно посылает ему запрос на выдачу следующих двадцати пакетов и т. д.

Данный способ передачи (Go Back n ARQ) выгодно отличается от ранее описанного (Stop and Wait ARQ) тем, что, в отсутствие ошибок, приостановки выдачи потока данных происходят после приёма группы из n пакетов, а не после приёма каждого из них. Основной недостаток — при искажении только одного пакета повторно передаётся целая группа.

Существуют и другие способы, при которых повторно передаются только те пакеты, в которых обнаружены ошибки. При этом задача восстановления принятого массива пакетов решается приёмным узлом. Он реконструирует массив вставкой в него повторно принятых пакетов. Это несколько усложняет логику работы приёмного узла и увеличивает объём его буферной памяти.

Предлагаемый способ повторной передачи ошибочных пакетов по компьютерной сети

Идея способа [1] состоит в том, что передающий узел ведёт непрерывную передачу пакетов. Получение подтверждения приёма пакета k (ACKk) означает, что этот пакет и все предыдущие приняты правильно; поэтому их копии в буферной памяти передающего узла уничтожаются. В случае получения неподтверждения приёма пакета с номером j (NAKj) передающий узел продолжает непрерывную передачу, возвращаясь к пакету с номером j и следующими за ним. Поясним сказанное с использованием сценария передачи, отражённого на рис. 3.

Узлы A и B содержат блоки буферной памяти. Как показано на рисунке, буферная память типа FIFO (First Input — First Output) передающего узла A объёмом 64 пакета представлена кольцом из 64 ячеек. Ячейки пронумерованы от 0 до 63. Символами Y, Z, V и W обозначены диаграммы состояния буферной памяти узла A на различных стадиях процесса передачи данных. Буферная память узла B не представляет особого интереса при дальнейшем описании, поэтому диаграммы её заполнения на рис. 3 не показаны.

В исходном состоянии буферная память узла A пуста (диаграмма Y). Предположим, что из узла A в узел B необходимо передать некоторый массив данных, представленный последовательностью пакетов. После установления сеанса связи между узлами A и B события развиваются следующим образом.

Буферная память узла A заполняется пакетами для передачи. Одновременно с этим начинается выдача пакетов в узел B. В момент, соответствующий диаграмме Z, в памяти узла A хранятся N пакетов; из них пакеты с номерами 0 — 26 уже отправлены в узел B, пакет 27 находится в процессе передачи, остальные (28 — N) ждут отправки. В данном примере нумерация пакетов совпадает с нумерацией ячеек памяти, в которых они хранятся.

В процессе передачи пакета 27 (диаграмма V) узел B выдаёт в обратном направлении ответный пакет ACK23. Как отмечалось, ответный пакет несёт информацию о том, что пакеты 0 — 23 успешно приняты узлом B, следовательно, их можно уничтожить в буферной памяти узла A, освободив соответствующие ячейки для размещения в них новых паке-

тов. Из диаграммы V следует, что после уничтожения пакетов 0 — 23 свободная область буферной памяти узла А простирается от ячейки N + 1 до ячейки 23, включая ячейку 0.

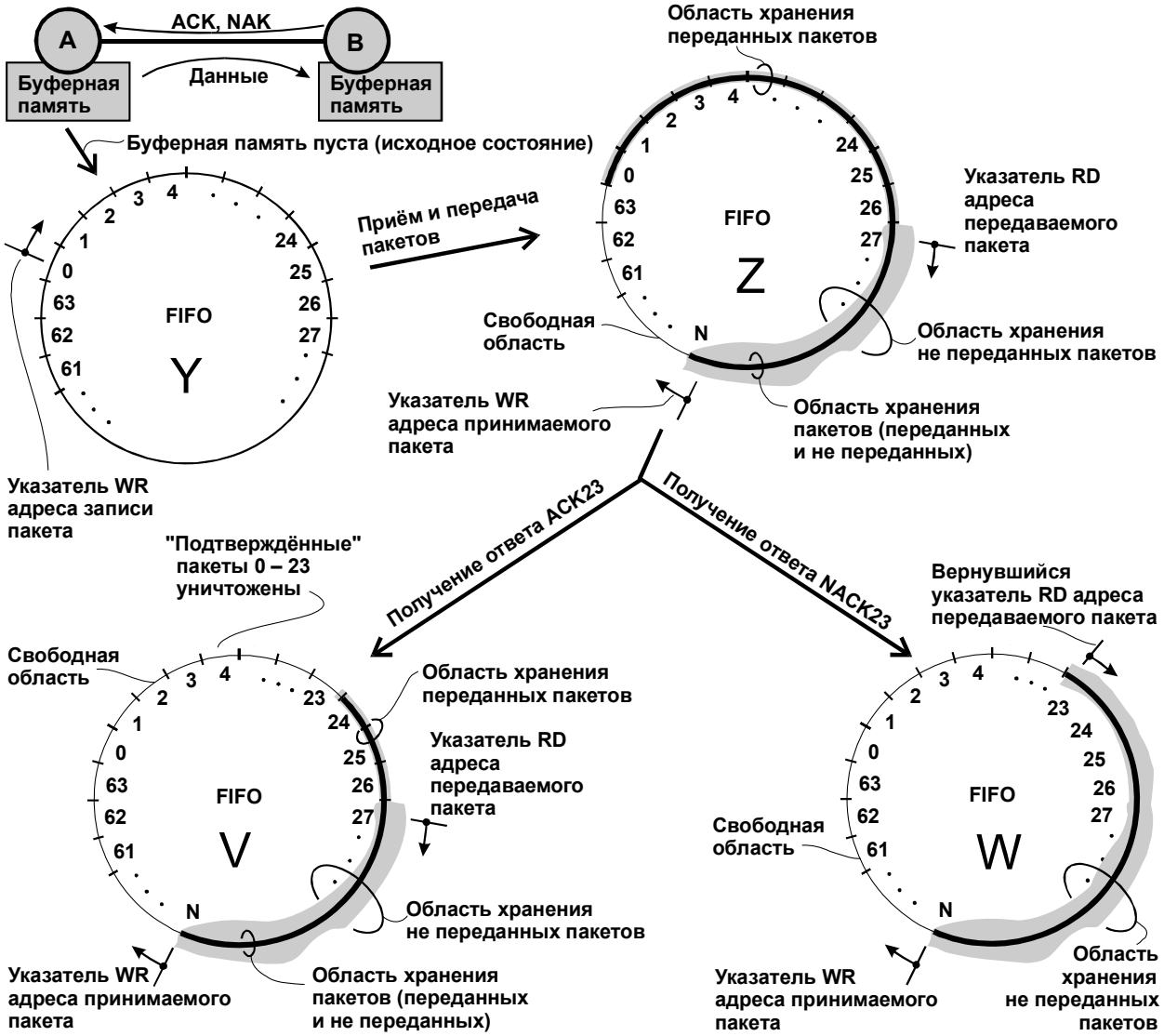

Рис. 3. К пояснению идеи предлагаемого способа взаимодействия передающего и приёмного узлов сети: динамика заполнения буферной памяти передающего узла

Вернёмся к диаграмме Z и рассмотрим иное развитие событий, когда из узла В в узел А прибыл пакет NAK23 (диаграмма W), означающий, что в пакете с номером 23 обнаружена некорректируемая ошибка. Узел А реагирует на это событие следующим образом.

Во-первых, он уничтожает в своей выходной буферной памяти пакеты с номерами 0 — 22, так как получение пакета NAK23, содержащего информацию о номере ошибочного пакета, дополнительно свидетельствует о том, что все предыдущие пакеты (0 — 22) приняты без ошибок. Таким образом, незанятая область выходной буферной памяти мгновенно увеличивается на 23 ячейки. Во-вторых, узел А прекращает выдачу пакета 27 (диаграмма Z) и возвращается к выдаче пакета 23 и всех последующих (диаграмма W).

Итак, узел А передаёт пакеты без пауз и, при необходимости, возвращается к передаче того пакета, который либо был потерян в пути, либо отвергнут узлом В из-за (*а*) содержащейся в нём некорректируемой ошибки или (*б*) в связи с переполнением входной буферной памяти. Поясним последнюю ситуацию (*б*).

Если буферная память узла В неспособна принять пакет с номером i из-за отсутствия хотя бы одной свободной ячейки, то этот пакет теряется. В этом случае узел А получает от узла В ответный пакет NAKi и вновь пытается передать пакет i. Но если буферная

память узла B по-прежнему отвергает его, то пакет i вновь теряется, узел A вновь получает ответный пакет NAKi и т. д. Такое состояние системы длится до тех пор, пока буферная память узла B остаётся полностью заполненной. После частичного или полного освобождения буферной памяти узла B очередной ответный пакет NAKi не посылается, передача пакетов возобновляется, начиная с пакета i.

В такой ситуации, чтобы не “засорять” трафик ненужными многократными встречными пересылками пакетов i и NAKi, желательно как можно раньше прервать эти пересылки приостановкой работы передатчика узла A до момента частичного или полного освобождения буферной памяти узла B (по усмотрению этого узла). Приостановку можно осуществлять двумя способами.

Первый способ приостановки работы передатчика узла A. Из узла B в узел A пересылается некоторый управляющий пакет XOFF (команда “Стоп”) по аналогии с известными способами программного управления потоками данных [2, 3]. После частичного или полного освобождения буферной памяти узла B он посылает в узел A команду (пакет) NAKi, в соответствии с которой этот узел возобновляет передачу пакетов, начиная с пакета i.

Второй способ приостановки работы передатчика узла A. Узел B не оповещает узел A о переполнении своей буферной памяти, точнее, он не посылает узлу A вообще никаких ответных пакетов. В буферной памяти узла A при этом не происходит освобождения ячеек, приём которых подтверждён узлом B. Поэтому буферная память узла A постепенно заполняется до предела, после чего её дальнейшего пополнения не происходит, а передатчик автоматически выключается. После частичного или полного рассасывания пакетов из буферной памяти узла B этот узел посылает узлу A команду NAKi, в соответствии с которой ранее выключенный передатчик возобновляет пересылку пакетов, начиная с пакета i.

Таким образом, наряду с повторной передачей пакетов осуществляется управление потоком данных.

Дополнительные преимущества предлагаемого способа передачи пакетов проявляются в цепях, содержащих несколько промежуточных узлов (рис. 4).

Рис. 4. Передача пакетов данных через несколько узлов

Данные из узла A в узел B передаются через промежуточные узлы (например, маршрутизаторы) M и N. Блоки буферной памяти узлов A и B являются, соответственно, выходным и входным. Блоки буферной памяти узлов M и N являются одновременно входными и выходными по отношению к потоку пакетов, передаваемых из узла A в узел B. Если узлы M и N, так же как и узлы A и B, способны выполнять описанные ранее действия по управлению потоком данных и исправлению ошибок повторной передачей пакетов, то можно полагать, что вся цепь передачи данных состоит из трёх “почти независимых” подсистем: A — M, M — N и N — B.

Исправление ошибок, возникающих в подсистемах, и управление соответствующими потоками данных осуществляется параллельно. Так, узлу A не нужно ждать, пока ошибка, возникшая в линии связи с узлом M будет распознана оконечным узлом B, как это обычно делается. Эта ошибка будет исправлена “на месте” подсистемой A — M в то время как другие подсистемы могут параллельно исправлять другие ошибки.

Заключение

В статье рассмотрены традиционные и предлагаемый [1] способы исправления ошибок передачи данных, использующие повторную пересылку ошибочных пакетов. В отличие от известных способов, передающий узел использует “оптимистическую” тактику передачи пакетов, которая состоит в том, что исходящий поток данных не прекращается “в надежде” на то, что все пакеты расцениваются приёмным узлом как безошибочные. При получении информации об ошибке передающий узел мгновенно возвращается к соответствующей точке повтора и продолжает непрерывную выдачу потока пакетов.

Такой способ позволяет уменьшить затраты времени, связанные с повторными передачами пакетов. Кроме того, решается задача управления потоком данных. При наличии нескольких промежуточных узлов на трассе передачи пакетов процессы управления потоком данных и исправления ошибок на разных участках этой трассы протекают независимо и одновременно.

ЛИТЕРАТУРА:

1. Пат. США № 6.760.307 B2 http://www.uspto.gov.

2 Б.В. Шевкопляс. Особенности управления потоками данных в интерфейсах семейства RS. — Схемотехника, 2004, № 9, с. 34 — 36, № 10, с. 34 — 36. Электронная версия статьи: http://lit.lib.ru/s/shewkopljas_b_w/.

3. Б.В. Шевкопляс. Вероятностная синхронизация в телекоммуникационных системах: вставка команд в поток данных без использования избыточных битов. — Схемотехника, 2005, № 5, с. 23 — 25, № 6, с. 23 — 26. Электронная версия статьи: http://lit.lib.ru/s/shewkopljas_b_w/.

Б. Шевкопляс