Исследование передачи коротких видеопотоков MPEG-DASH в сетях Wi-Fi*

Красилов А.Н.
ИППИ РАН
krasilov@iitp.ru

Любогощев М.В.
ИППИ РАН
lyubogoshchev@phystech.edu

Хоров Е.М.
ИППИ РАН
khorov@iitp.ru

Аннотация

В данной работе рассматривается передача видеопотоков MPEG-DASH в сетях Wi-Fi. Согласно стандарту MPEG-DASH пользователь может адаптивно изменять битрейт (качество) загружаемого видеопотока с течением времени. Однако стандартом не регламентирован алгоритм выбора битрейта. В работе рассмотрены различные алгоритмы адаптивного выбора битрейта и показано, как качество восприятия видеоизображения пользователем зависит от выбранного алгоритма и его параметров. В частности показано, что для коротких видеофрагментов средний битрейт загруженного видеопотока оказывается значительно меньше доступной пропускной способности канала. Проанализированы причины возникновения данной проблемы и возможные подходы к ее решению.

1. Введение

На сегодняшний день доля видеотрафика в Интернете составляет около 60% и, согласно прогнозам компании Cisco Systems, вырастет до 80% к 2020 году [1]. Кроме того, растет доля мобильного трафика и доля видеоформатов HD и Ultra HD. Отличительной особенностью мобильных (беспроводных) устройств является то, что скорость передачи данных может существенно меняться со временем. Это обусловлено, во-первых, свойствами среды передачи: изменением силы принимаемого сигнала при движении устройства, наличием случайных помех; а во-вторых, переменным числом устройств, соревнующихся за беспроводной канал. В то же время, при передаче видеопотоков необходимо поддерживать скорость передачи данных большую заданного порога, зависящего от разрешения (качества) просматриваемого видеопотока.

Данная проблема была частично решена с помощью механизма буферизации, т.е. заблаговременной загрузки на устройство некоторой части видеофайла, позволяющего сгладить кратковременные ухудшения скорости соединения. С другой стороны, пользователи далеко не всегда досматривают видеофайл до конца, поэтому не имеет смысла загружать на устройство весь видеофайл, а достаточно поддерживать объем буфера на определенном уровне. Этот подход получил название Progressive Download: после стартовой фазы накопления буфера на стороне пользователя (англ. Initial burst), серверу достаточно отправлять данные клиенту со скоростью, равной скорости воспроизведения.

В связи с тем, что в беспроводных сетях пропускная способность соединения может существенно изменяться со временем, перед началом передачи видеопотока невозможно выбрать некоторое фиксированное разрешение загружаемого видеопотока так, чтобы при этом использовать всю доступную пропускную способность соединения, и избежать прерываний воспроизведения из-за опустошения буфера. Таким образом, возникает задача адаптивного выбора разрешения просматриваемого видеопотока, удовлетворяющего текущим свойствам канала передачи данных. Для решения этой задачи был разработан международный стандарт адаптивной потоковой передачи данных MPEG-DASH (Dynamic Adaptive Streaming over HTTP) [2].

Стандарт MPEG-DASH определяет протокол взаимодействия клиента и сервера при передаче видеопотока. Видеофайл на сервере разбит на короткие сегменты. Для каждого сегмента на сервере хранятся его копии в различных разрешениях. Клиент на основании некоторого алгоритма определяет разрешение следующего загружаемого сегмента. Стандарт описывает протокол взаимодействия клиента и сервера, но не определяет конкретного алгоритма принятия решений для клиента, предоставляя разработчикам возможность выбрать алгоритм, наиболее подходящий для решения конкретной задачи. В связи с этим с момента начала разработки стан-

*Исследование выполнено в ИППИ РАН за счет гранта Российского научного фонда (проект № 16-19-10687)

дарта в литературе было представлено множество различных подходов к реализации этого алгоритма. Некоторые из этих подходов исследованы в работе [3]. Кроме того, производителями програмного обеспечения представлен ряд проприетарных алгоритмов, в том числе реализованных в мультимедийных проигрывателях Microsoft Smooth Streaming, Apple HTTP Live Streaming (HLS), и в открытом проигрывателе от Adobe Systems: Open Source Media Framework (OSMF). Данные алгоритмы исследовались в работе [4].

В упомянутых выше работах исследования проводились в условиях заранее заданой, пусть и переменной, пропускной способности канала. Кроме того, не учитывалось наличие другого сетевого трафика, например веб-трафика, и его взаимодействие с видеопотоками MPEG-DASH. В данной работе рассматривается передача видеопотоков в сети Wi-Fi, в том числе, при наличии фонового веб-трафика, и исследуется работа двух алгоритмов, использующих различные подходы при выборе разрешения. Выбор технологии Wi-Fi обусловлен тем, что на сегодняшний день в сетях, использующих данную технологию, генерируется около 40% трафика и эта доля продолжает расти [1]. В работе подробно исследуется сценарий, в котором DASH-клиент не смотрит видеофайл целиком, а ищет интересные ему фрагменты. Иными словами, он регулярно прерывет воспроизведение и запрашивает новый еще не загруженный видеофрагмент. Подобное поведение клиента является достаточно распространенным. Например, согласно исследованиям компании Visible Measures до 20% всех просмотров видеофайлов пользователями имеют продолжительность менее 10 секунд [5]. Кроме того, широкое распространение получили такие сервисы, как Vine и Video on Instagram, позволяющие пользователям обмениваться короткими видеофайлами в социальной сети. В данной статье проанализированы особенности работы различных алгоритмов выбора разрешения видеопотока в описанном выше сценарии, описаны возникающие при этом проблемы и предложены пути их решения.

Дальнейшее изложение работы построено следующим образом. В разделе 2 представлено краткое описание стандарта MPEG-DASH. В разделе 3 описаны расмотренные нами алгоритмы выбора разрешения видеопотока. В разделе 4 с помощью среды имитационного моделирования NS-3 проведено исследование данных алгоритмов. Раздел 5 содержит выводы и направления дальнейших исследований.

2. Стандарт MPEG-DASH

Стандарт MPEG-DASH описывает протокол взаимодействия сервера, предоставляющего видеоконтент (видеофайлы), и клиента, загружающего этот контент посредством протокола HTTP. В качестве нижележащего протокола транспортного уровня используется протокол TCP, гарантирующий передачу данных без потерь и дубликатов. Согласно стандарту видеоконтент на сервере разбит на сегменты длительностью τ. Сегменты на сервере хранятся в нескольких битрейтах. Под битрейтом видеосегмента будем понимать отношение размера сегмента в битах к продолжительности его воспроизведения. Больший битрейт подразумевает лучшее качество видеопотока.

Для описания битрейтов, доступных для каждого сегмента, на сервере хранится файл описания видеопотока (англ. Media Presentation Description, MPD). После установления TCP-соединения клиент запрашивает у сервера этот файл, и далее, по мере воспроизведения видеофрагмента, на основе информации, представленной в файле, и некоторого алгоритма, который не регламентируется стандартом, клиент адаптивно выбирает, когда и в каком разрешении загружать следующий сегмент.

3. Алгоритмы выбора битрейта

Алгоритм принятия решений на стороне клиента должен быть нацелен на достижение максимального качества видеоизображения, просматриваемого пользователем (англ. Quality of User Experience, QoE). Согласно [6] на QoE оказывают влияние следующие факторы:

1. ожидание начала воспроизведения видеофрагмента;
2. число и продолжительность пауз при воспроизведении видеофрагмента;
3. разрешение просматриваемого видеофрагмента;
4. частота изменений разрешения и их амплитуда.

В данном разделе нами рассматриваются примеры двух алгоритмов, использующих различные подходы к выбору битрейта. Основное отличие этих алгоритмов заключается в том, используется ли при принятии решения о выборе битрейта информация о текущем состоянии буфера.

3.1. Алгоритм Instant

Одним из простейших алгоритмов выбора битрейта является алгоритм, представленный в работе [7]. Аналогично [3] будем называть его Instant, в силу специфики его работы. Данный алгоритм выбирает битрейт каждого последующего сегмента чуть меньшим пропускной способности соединения, измеренной за некоторый промежуток времени. В алгоритме используются 3 параметра: (1) минимальный

объем буфера b_{min} (в секундах), при котором начинается воспроизведение видеофрагмента; (2) коэффициент чувствительности алгоритма β ($\beta \leq 1$), определяющий соотношение между измеренной пропускной способностью и выбранным битрейтом; (3) длительность временного интервала Δ, за который усредняется наблюдаемая пропускная способность соединения с сервером.

Суть алгоритма заключается в следующем. После окончания загрузки i-го сегмента оценивается средняя пропускная способность соединения при загрузке этого сегмента:

$$\rho_s(i) = \frac{r(i)\tau}{t_f(i) - t_s(i)}, \quad (1)$$

где $r(i)$ – битрейт i-го сегмента, а $t_s(i)$ и $t_f(i)$ – времена начала и окончания загрузки сегмента соответственно.

Средняя пропускная способность соединения в интервале от t_1 до t_2 определяется, как

$$\rho(t_1,t_2) = \frac{\sum_i \rho_s(i)|[t_s(i),t_f(i)] \cap [t_1,t_2]|}{\sum_i |[t_s(i),t_f(i)] \cap [t_1,t_2]|},$$

где $|[x_1,x_2]| = x_2 - x_1$ — длительность интервала $[x_1,x_2]$.

Решение о выборе битрейта сегмента $i+1$ принимается следующим образом. Воспроизведение видеофрагмента на стороне клиента не начинается, пока объем буфера меньше величины b_{min}. Чтобы уменьшить задержку начала воспроизведения, при объеме буфера $b < b_{min}$ битрейт сегмента $i+1$ выбирается равным минимально возможному битрейту:

$$r(i+1) = r_{min} = \min_{r \in \Re} r,$$

где $\Re$ — множество доступных на сервере битрейтов согласно MPD.

Во всех остальных случаях выбирается максимально возможный битрейт, не превосходящий оценку пропускной способности $\rho(t-\Delta_\rho,t)$, умноженной на коэффициент β:

$$r(i+1) = \max_{r < \beta\rho(t-\Delta,t)} r, \quad r \in \Re.$$

3.2. Алгоритм Миллера

Другой рассматриваемый нами алгоритм предложен в работе [6]. Далее будем называть его алгоритмом Миллера по фамилии первого автора. Данный алгоритм стремится поддерживать объем буфера в заданных границах. Алгоритм понижает битрейт, если объем буфера стал меньше нижней границы b_{low}, и повышает, если накоплен буфер некоторого объема и объем буфера растет достаточно быстро. Данный алгоритм меняет битрейт пошагово, т.е.

$$r(i+1) \in \left\{r^{\uparrow}(i), r(i), r^{\downarrow}(i)\right\},$$

где $r^{\uparrow}(i)$ ($r^{\downarrow}(i)$) – следующий битрейт из $\Re$, больший (меньший) текущего битрейта $r(i)$, если такой есть. Кроме того, обозначим максимальный и минимальный битрейт из $\Re$, как r_{max} и r_{min} соответственно.

Алгоритм использует следующий набор параметров: (а) пороговые значения объема буфера $b_{min} \leq b_{low} < b_{high}$; (б) коэффициенты чувствительности алгоритма $\alpha_1, ..., \alpha_5$; (в) длительность Δ интервала сглаживания пропускной способности. При выборе битрейта следующего запрашиваемого сегмента алгоритм получает на вход динамику изменения объема буфера $b(t)$ и статистику пропускной способности соединения при загруке предыдущих сегментов $\rho_s(i)$, определяемую по формуле (1). Работа алгоритма делится на 2 фазы: фаза начальной загрузки буфера и фаза нормальной работы.

Основная задача фазы начальной загрузки буфера — обеспечить баланс между сокращением стартовой задержки воспроизведения, накоплением достаточного объема буфера, чтобы избежать пауз воспроизведения, и соответствием между выбранным битрейтом видеопотока и доступной пропускной способностью соединения. Выбор битрейта в начальной фазе зависит от текущего объема буфера $b(t)$.

- Если $b(t) \leq b_{min}$ и $r^{\uparrow}(i) \leq \alpha_2\rho(t-\Delta,t)$, то $r(i+1) = r^{\uparrow}(i)$.
- Если $b_{min} < b(t) \leq b_{low}$ и $r^{\uparrow}(i) \leq \alpha_3\rho(t-\Delta,t)$, то $r(i+1) = r^{\uparrow}(i)$.
- Если $b(t) > b_{low}$ и $r^{\uparrow}(i) \leq \alpha_4\rho(t-\Delta,t)$, то $r(i+1) = r^{\uparrow}(i)$.
- Если $b(t) \geq b_{high}$, то запрос следующего сегмента откладывается, пока объем буфера не станет меньше b_{high}. Уменьшение объема буфера произойдет за счет того, что пользователь просмотрит часть видеофрагмента.
- Во всех остальных случаях алгоритм не изменяет битрейт $r(i+1) = r(i)$.

Коэффициенты $\alpha_2, \alpha_3, \alpha_4$ подобраны так, что с увеличением объема буфера растет агрессивность алгоритма по отношению к выбору битрейта (см. табл. 1). Начальная фаза длится, пока выполняются все следующие условия:

- текущий битрейт меньше средней пропускной способности $r(i) \leq \alpha_1\rho(t-\Delta_\rho,t)$;
- продолжается рост объема буфера $b(t_2) \geq b(t_1)$, при $t_2 > t_1$;
- не достигнут максимально доступный битрейт r_{max}.

Таблица 1. Параметры алгоритмов выбора битрейта

Instant	β	0,95
	Δ, с	10
Алгоритм Миллера	b_{low}, с	10
	b_{high}, с	50
	α_1	0,75
	α_2	0,33
	α_3	0,50
	α_4	0,75
	α_5	0,90
	Δ, с	10

Задачей фазы нормальной работы является поддержание буфера на уровне $b_{opt} = \frac{b_{low}+b_{high}}{2}$ и изменение битрейта в соответствии с текущим объемом буфера.

- Если $b(t) < b_{min}$, то выбирается наименьший из доступных битрейтов, $r(i+1) = r_{min}$.
- Если $b_{min} < b(t) \le b_{low}$ и $r(i) \ge \rho_s(i)$, то $r(i+1) = r^{\downarrow}(i)$.
- Если $b(t) > b_{opt}$, но $r^{\uparrow}(i) > \alpha_5 \rho(t-\Delta_\rho, t)$, то клиент не запрашивает новый сегмент, пока $b(t) > b_{opt}$.
- Если $b(t) > b_{high}$ и $r^{\uparrow}(i) \le \alpha_5 \rho(t-\Delta_\rho, t)$, то $r(i+1) = r^{\uparrow}(i)$.
- Во всех остальных случаях алгоритм не изменяет битрейт $r(i+1) = r(i)$.

4. Численные результаты

4.1. Постановка экспериментов

Проведем исследование рассмотренных выше алгоритмов выбора битрейта с помощью среды имитационного моделирования NS-3 [8]. Рассмотрим следующий сценарий (см. рис. 1). Веб-сервер соединен с точкой доступа с помощью проводного соединения пропускной способностью 1 Гбит/с и задержкой распространения сигнала из конца в конец RTT=10 мс. Точка доступа сети Wi-Fi и клиентские устройства на физическом уровне используют протокол IEEE 802.11a. В качестве алгоритма управления скоростью передачи используется алгоритм Minstrel. Клиентские устройства расположены на расстоянии 36 метров от точки доступа. Как показали предварительные эксперименты, при таком расположении устройств суммарная пропускная способность на уровне приложений $S_0 = 12,7$ Мбит/с.

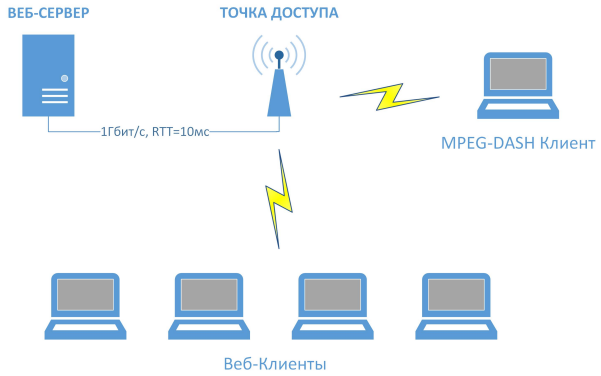

Рис. 1. Топология сети

Клиенты генерируют следующий трафик. Одно устройство является DASH-клиентом, просматривающим видеофайлы с сервера. Остальные $N_{web} = 4$ клиентских устройства загружают с сервера веб-страницы через экспоненциально распределенное время с заданным средним $T_{interReq}$. Размер загружаемых веб-страниц выбирается таким образом, чтобы в среднем на N_{web} устройств приходилась фиксированная доля пропускной способности канала $Load$:

$$L = \frac{S_0 \cdot T_{interReq} \cdot Load}{N_{web}}.$$

DASH-клиент просматривает видеофайлы следующим образом. Через случайное время с заданным распределением воспроизведение текущего видеофрагмента завершается и начинается загрузка нового видеофрагмента. Таким образом, вместо того чтобы смотреть видеофайл от начала и до конца, клиент ищет наиболее интересные ему эпизоды. В данной работе время между просмотром двух последовательных видеофрагментов имеет экспоненциальное распределение со средним $T_{interClick}$, которое ограничено снизу длительностью одного сегмента τ.

Чтобы минимизировать задержку при переключении на новый видеофрагмент и не загружать в буфер пакеты из предыдущего видеофрагмента, TCP-соединение с сервером принудительно закрывается, и открывается новое соединение, с помощью которого начинается загрузка нового видеофрагмента.

Длительность эксперимента $T_{sim} = 1500$ с. Для каждого эксперимента проведены 5 независимых прогонов имитационной модели. Другие параметры эксперимента представлены в табл. 2.

Чтобы оценить качество восприятия видеоизображения клиентом, в каждом эксперименте измеряются следующие величины:

1. доля пропускной способности канала, приходящаяся на DASH-клиента и суммарная доля пропускной способности канала, приходящаяся на всех веб-клиентов;

Таблица 2. Параметры экспериментов

$T_{interReq}$, с	30
$\Re$, Мбит/с	0,3; 0,5; 1,0; 1,8; 2,5; 5,0; 8,0; 16,0; 24,0; 40,0
Load, %	28, 65
τ, с	2

2. средний битрейт видеопотока, просмотренного DASH-клиентом;
3. средний логарифм отношения битрейта просматриваемого сегмента к минимальному битрейту;
4. средняя задержка начала воспроизведения нового видеофрагмента;
5. общее число пауз за все время эксперимента;
6. суммарная продолжительность пауз за все время эксперимента;
7. среднее число изменений битрейта за 1 секунду просмотренного видеофрагмента;
8. средний модуль изменения битрейта.

4.2. Анализ результатов

На рисунках 2 и 3 представлениы результаты имитационного моделирования для двух исследуемых алгоритмов при значениях параметра $b_{min} = \{2;5\}$ с и значениях нагрузки, генерируемой веб-клиентами, $Load = \{28;65\}$%.

Штриховой линией на левом верхнем графике показана суммарная доля пропускной способности канала, приходящаяся на веб-клиентов. Можно видеть, что величина, измеренная в эксперименте, оказалась близкой к изначально заданной нагрузке в 28 или 65% соответственно. Сплошной линией на том же графике показана доля пропускной способности, приходившаяся на DASH-клиента. Из полученных результатов следует, что DASH-клиенту не удается полостью использовать оставшуюся пропускную способность канала. Причины данного эффекта заключаются в следующем. Во-первых, для обоих алгоритмов пропускная способность соединения может быть недоиспользована в связи с задержкой определения доступной пропускной способности соединения протоколом TCP, а именно, работой механизма TCP Slow Start. Особенно значительно влияние этого фактора при малых значениях $T_{interClick}$ (в этом случае DASH-клиент часто устанавливает новое соединение) и малой нагрузке, генерируемой веб-клиентами. Во-вторых, недостаточное использование пропускной способности алгоритмом Миллера объясняется дискретностью доступного набора битрейтов и тем, что алгоритм прекращает загружать сегменты после достижения определенного объема буфера.

Из графика среднего битрейта воспроизведенных видеофрагментов можно видеть, что на коротких видеофрагментах (при малых значениях $T_{interClick}$) средний битрейт значительно меньше доступной DASH-клиенту пропускной способности канала. Основной причиной данного эффекта является недостаток информации о состоянии канала в начале передачи видеофрагмента. Если сравнивать два алгоритма выбора битрейта, то можно видеть, что средний битрейт выше у алгоритма Instant. Это объясняется тем, что данный алгоритм выбирает битрейт следующего сегмента в соответствии с измеренной пропускной способностью соединения вне зависимости от битрейта предыдущего загружаемого сегмента или текущего состояния буфера. В то время как алгоритм Миллера при малом объема буфера повышает битрейт пошагово и только тогда, когда измеренная пропускная способность канала значительно больше следующего битрейта. Кроме того, из представленных графиков видно, что уменьшение величины b_{min} позволяет увелчить средний битрейт, так как в этом случае алгоритмы раньше начинают повышать битрейт запрашиваемых сегментов.

Из графиков начальной задержки можно видеть, что использование $b_{min} = 2$ с позволяет также снизить задержку начала воспроизведения видеофрагмента. Это обусловлено тем, что при меньшем значении b_{min} воспроизведение начинается после загрузки меньшего числа сегментов. С другой стороны, график суммарной продолжительности и числа пауз показывает недостаток алгоритма Instant при $b_{min} = 2$ с. Он обусловлен агрессивной политикой повышения битрейта, реализуемой на начальном этапе воспроизведения видеофрагмента, особенно уязвимом к флуктуациям пропускной способности соединения. Так как в начале воспроизведения объем буфера $b = b_{min}$, то высока вероятность опустошения буфера и возникновения пауз при воспроизведении. Алгоритм Миллера, в свою очередь, допускает паузы в воспроизведении при росте нагрузки на сеть со строны веб-клиентов и увеличении средней продолжительности видеофрагмента. Это связано с тем, что в отличие от алгоритма Instant данный алгоритм поддерживет ограниченный объем буфера, что в случае значительного и долговременного уменьшения пропускной способности соединения приводит к опустошению буфера и наступлению пауз. Большее число пауз у алгоритма Миллера при $b_{min} = 2$ с, по сравлнению с $b_{min} = 5$ вызвано слишком поздним переключением на минимальный доступный битрейт.

Также на представленных графиках можно видеть различия алгоритмов в частоте и амплитуде

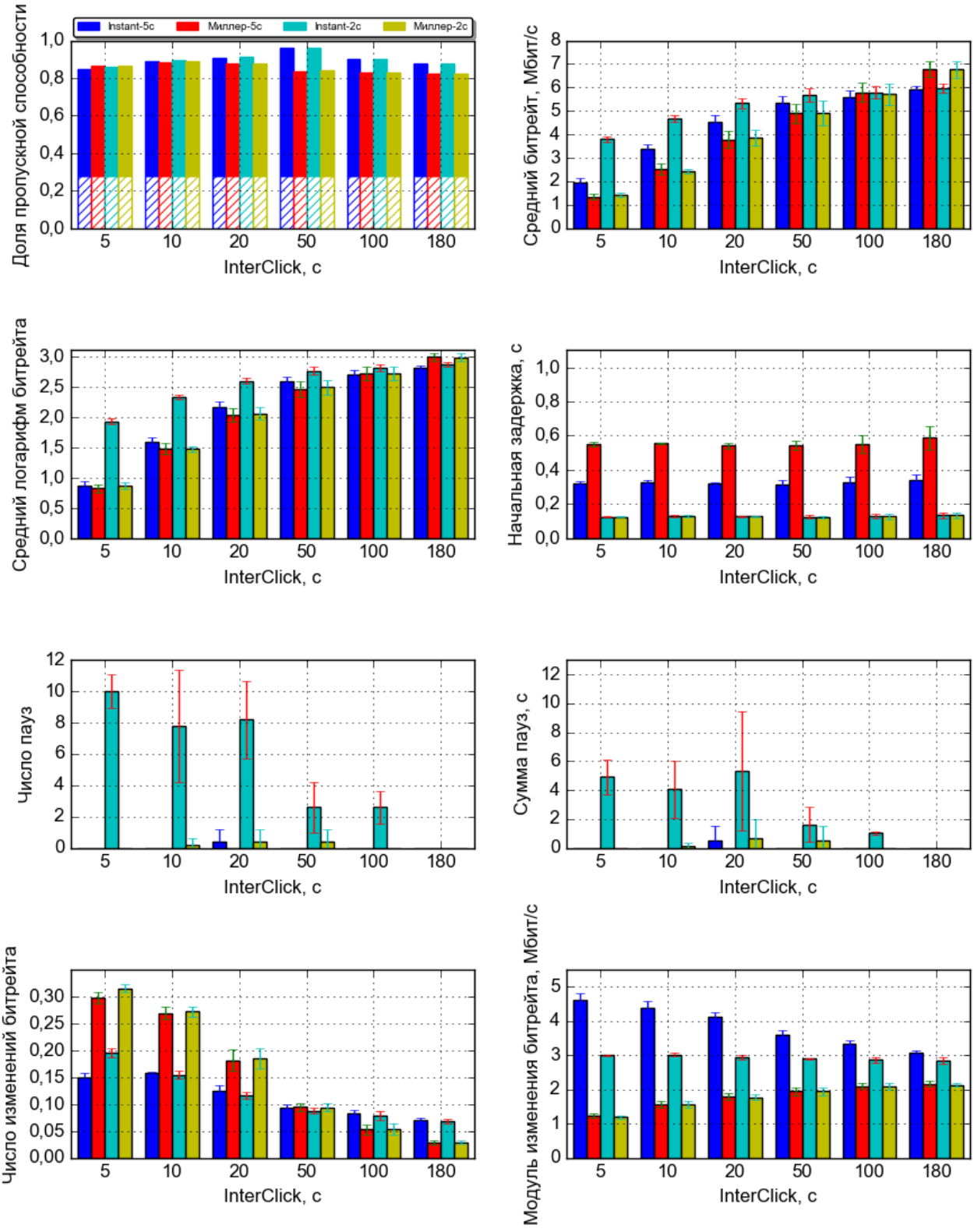

Рис. 2. Результаты моделирования для $Load = 28\%$

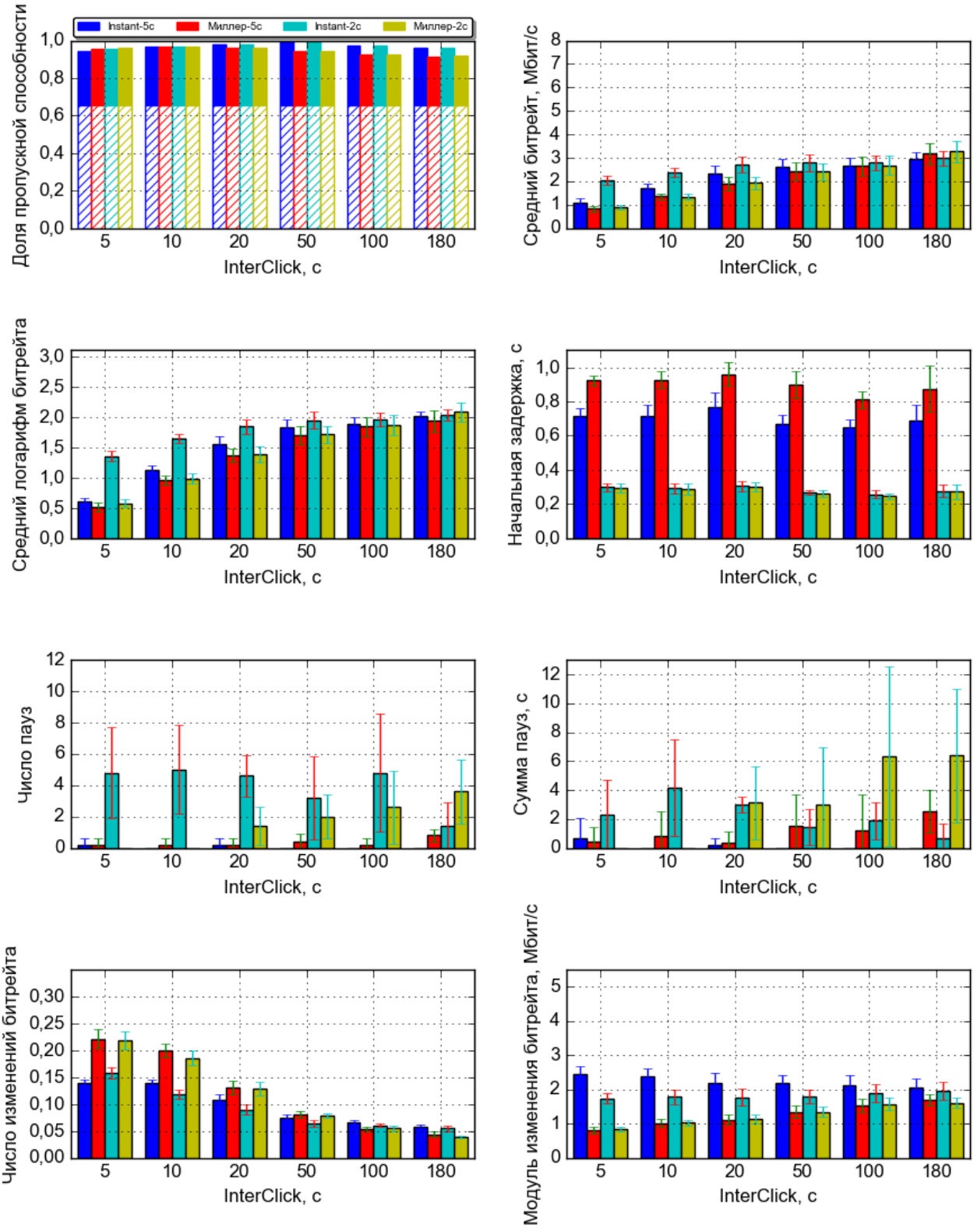

Рис. 3. Результаты моделирования для $Load = 65\%$

изменений битрейта. На коротких видеофрагментах Instant реже меняет битрейт и на большее по модулю значение, а алгоритм Миллера – чаще, но на меньшее значение. Причина данного эффекта заключается в том, что алгоритм Миллера изменяет битрейт пошагово, а Instant сразу выбирает битрейт, удовлетворяющий текущей пропускной способности соединения. В то же время на длинных видеофрагментах алгоритм Миллера показывает лучшие результаты по обоим показателям: он реже меняет битрейт и меняет его на меньшую величину. Таким образом, алгоритм Миллера оказывается более стабильным на длинных видеофрагментах.

5. Заключение

Стандарт MPEG-DASH определяет протокол передачи видеопотока от сервера клиенту, позволяющий адаптивно изменять битрейт загружаемого видеопотока в соответствии с текущими свойствами канала передачи данных. В данной работе рассматривается сценарий передачи видеопотока с использованием технологии Wi-Fi при наличии в сети фонового веб-трафика. При этом пользователь последовательно просматривает несколько видеофрагментов. Было проанализировано влияние используемого алгоритма выбора битрейта и его параметров на качество восприятия видеоизображения пользователем.

Результаты исследования показывают, что для коротких видеофрагментов средний битрейт просматриваемого пользователем видеоизображения оказывается существенно меньше доступной пропускной способности соединения. Это вызвано недостатком информации о пропускной способности соединения в начале загрузки видеофрагмента. Для решения данной проблемы можно использовать следующие подходы.

1. Использовать статистику пропускной способности, собранную при передаче предыдущих видеофрагментов.

2. Реализовать кросс-уровневый протокол, позволяющий передавать информацию о пропускной способности соединения с канального уровня на уровень приложений.

Вместе с тем, как показывает проведенное в данной работе исследование, увеличение битрейта в начале передачи видеопотока может приводить к следующим проблемам. Во-первых, увеличение битрейта сегментов, загружаемых до начала воспроизведения видеофрагмента, приводит к увеличению задержки начала воспроизведения. Во-вторых, более агрессивная политика выбора битрейта сегментов при малом объеме буфера может приводить к паузам в воспроизведении.

Указанные проблемы могут быть решены, в частности, за счет приоритизации на канальном уровне трафика для DASH-клиентов, начинающих просмотр видеофайлов или клиентов с малым объемом буфера. Более подробное исследование обозначенных способов улучшения качества восприятия видеоизображения пользователем является направлением дальнейшей работы.

Список литературы

[1] Cisco Visual Networking Index: Forecast and Methodology, 2015–2020 White Paper.

[2] MPEG-DASH 2nd Edition Specification (ISO/IEC 23009-1:2014).

[3] C. Timmerer, M. Maiero, B. Rainer *Which Adaptation Logic? An Objective and Subjective Performance Evaluation of HTTP-based Adaptive Media Streaming Systems.*

[4] Saamer Akhshabi, Ali C. Begen, and Constantine Dovrolis. 2011. *An experimental evaluation of rate-adaptation algorithms in adaptive streaming over HTTP.* In Proceedings of the second annual ACM conference on Multimedia systems (MMSys '11). ACM, New York, NY, USA, 157-168. `http://doi.acm.org/10.1145/1943552.1943574`.

[5] *Visible Measures* `http://www.visiblemeasures.com/category/research/#`.

[6] K. Miller, E. Quacchio, G. Gennari, and A. Wolisz. *Adaptation Algorithm for Adaptive Streaming over HTTP. In Packet Video Workshop (PV)*, 2012 19th International, pages 173–178, May 2012.

[7] L. R. Romero. *A Dynamic Adaptive HTTP Streaming Video Service for Google Android.* Master's thesis, Royal Institute of Technology (KTH), Stockholm, October 2011.

[8] *Network simulator NS-3* `https://www.nsnam.org/`.