Совершенствование метода измерения пропускной способности сетевых устройств на основе адаптивных алгоритмов поиска с учетом истории измерений

С. Е. Кублицкий

Санкт-Петербургский политехнический университет Петра Великого

stanleykub@mail.ru

К. К. Семенов

Санкт-Петербургский политехнический университет Петра Великого

semenov_kk@spbstu.ru

Аннотация. **В работе рассмотрена задача сокращения времени тестирования пропускной способности программно-аппаратных комплексов, предназначенных для обработки сетевого трафика, при выпуске новых версий их программного обеспечения. Выполнение стандартной процедуры по спецификации RFC 2544, основанной на применении бинарного поиска, занимает значительное количество времени на каждый из вариантов условий тестирования, которых достаточно много, что замедляет цикл разработки. Предложен ряд модифицированных алгоритмов поиска, опирающихся не только на констатацию факта присутствия или отсутствия потерь при передаче данных, но также на количественную оценку потерь и априорную информацию о результатах предыдущих выполненных измерений. Для определения наиболее удачного варианта выполнена разработка ряда алгоритмов на языке Matlab и проведено имитационное моделирование для выполнения сравнения. Результаты показывают сокращение трудоемкости тестирования пропускной способности программно-аппаратных комплексов на (27÷50)% при сохранении требуемой достоверности результатов тестирования.**

Ключевые слова: нагрузочное тестирование; измерение производительности; пропускная способность; оптимизация алгоритмов; адаптивный поиск; анализ временных рядов

I. Введение

Современная разработка программного обеспечения (ПО) в парадигме DevOps требует не только высокой степени автоматизации процессов сборки, тестирования и развертывания программного обеспечения, но и существенного сокращения времени между внесением изменений и их выпуском в эксплуатацию [1, 2]. Ключевым элементом подобного конвейерного подхода выступает непрерывная интеграция, в рамках которой автоматизированное тестирование позволяет быстро выявлять регресс качества продукта по сравнению с предыдущими версиями. Вместе с тем, в литературе отмечается [3, 4], что существенная длительность процедур тестирования значительно удлиняет цикл разработки и увеличивает время вывода новых версий программного обеспечения на рынок.

Для ПО программно-аппаратных комплексов (ПАК) обработки сетевого трафика одной из ключевых контролируемых метрик является пропускная способность (ПС) l_{max}, под которой понимается такой максимальный объем сетевого трафика, проходящего через ПАК за единицу времени, который может быть полностью обработан в темпе его поступления [5]. Доля ρ пакетов трафика, пропущенных без обработки, называется потерями [5]. Основным применяемым в настоящее время подходом к измерению пропускной способности такого ПО является алгоритм, изложенный в методике RFC 2544 [6] и основанный на бинарном поиске. В работе [7] представлена критика данного подхода и принятого в [6] порогового значения потерь (равного 0%). Отмечается, что на практике чаще используют ненулевой порог небольшой величины (обычно 0,01-0,1%).

Согласно [6] процедура определения значения l_{max} максимальной ПС сводится к бинарному поиску по значениям величины потерь ρ, измеренным для различных значений l нагрузки, подаваемой на тестируемый программно-аппаратный комплекс. Такой подход основан на том обстоятельстве, что зависимость $\rho(l)$ для ПАК обработки сетевого трафика носит характер монотонно невозрастающей функции. Основной недостаток такого метода заключается в его высокой трудоемкости [7]: каждое измерение значения ПС занимает на практике десятки секунд (формирование трафика, прогрев, собственно само измерение).

При частых выпусках обновлений программного обеспечения и широкой номенклатуры поддерживаемых программно-аппаратных комплексов это создает узкое место в конвейере непрерывной интеграции. Таким образом, есть потребность в ускорении оценки значения ρ_{max} максимальной пропускной способности для сокращения сроков и затрат на тестирование ПО соответствующих ПАК без потери качества.

II. Постановка задачи

Цель настоящей работы заключалась в разработке и исследовании алгоритма оценки значения l_{max} максимальной пропускной способности ПАК обработки сетевого трафика, позволяющего за минимально возможное время достоверно подтвердить, что для новой сборки ПО значение l_{max} не снизилось. Данный алгоритм должен обеспечивать:

- достоверность результатов оценки l_{max} при высокой вариативности результатов измерения ПС вследствие действия случайных факторов;

- низкие риски ложного обнаружения уменьшения значения l_{max} по сравнению с нормативным или отсутствия регистрации произошедшего снижения l_{max}.

Предполагается, что учет степени потерь и исторических данных о ранее выполненных измерениях позволит сократить общее количество выполняемых измерений по сравнению с классическим бинарным поиском [6].

III. Стандартный метод решения на основе бинарного поиска [6]

Для оценки относительной эффективности предлагаемых в работе методов и подходов к оценке l_{max} был реализован бинарный поиск в его стандартной форме [6]. Суть алгоритма предельно проста: заданный интервал локализации значения нагрузки, соответствующей максимальной пропускной способности ПАК для обработки сетевого трафика, последовательно делится пополам с отбрасыванием той половины, на которой отсутствуют потери. Процесс продолжается до достижения требуемой точности определения l_{max}. Данный алгоритм служит репером для сравнения. Ниже представлено его соответствующее описание.

Входные данные Алгоритма 1:

$l_{low}^{(0)}, l_{high}^{(0)}$ – границы диапазона значений нагрузки на ПАК для обработки сетевого трафика, в пределах которого достигается максимальная ПС;

$\Delta_\rho = 0{,}1\%$ – пороговое значение допустимых потерь;

Δ_l – предел требуемой абсолютной погрешности определения значения l_{max}.

Промежуточные переменные Алгоритма 1:

$l_{low}^{(n)}, l_{high}^{(n)}$ – нижняя и верхняя границы интервала локализации такой нагрузки l_{max}, при которой обеспечивается максимальная пропускная способность, определенные на n-й итерации алгоритма поиска;

$l^{(n)}$– значение нагрузки, выбранной для измерения на n-й итерации;

$\rho_n = \left(\rho\left(l^{(n)}\right) + \varepsilon_n\right)$ – результат измерения потерь при нагрузке $l^{(n)}$, где ε_n – случайная погрешность результата измерений.

Псевдокод Алгоритма 1.

1) Присвоить n значение, равное 1.
2) Вычислить $l^{(n)} = \frac{1}{2} \cdot \left(l_{low}^{(n-1)} + l_{high}^{(n-1)}\right)$.
3) Получить результат измерения значения ρ_n.
4) Если $\rho_n \le \Delta_\rho$, то присвоить $l_{low}^{(n)}$ значение $l^{(n)}$. В противном случае присвоить $l_{high}^{(n)}$ значение $l^{(n)}$.
5) Если $l_{high}^{(n)} - l_{low}^{(n)} \ge \Delta_l$, то присвоить n значение (n+1) и перейти к шагу 2. В противном случае выйти из процедуры, назначив $l_{\max} = \frac{1}{2} \cdot \left(l_{low}^{(n)} + l_{high}^{(n)}\right)$.

IV. Предлагаемые алгоритмы поиска

Для уменьшения числа измерений, необходимых для определения значения l_{max}, был рассмотрен ряд других алгоритмов организации поиска, для каждого из которых на языке программирования R была составлена программная реализация. Ниже представлены описания рассмотренных методов.

4.1. *Адаптивный поиск с весовой функцией потерь*

Данный алгоритм расширяет возможности бинарного поиска за счет учета величины превышения порога потерь. При катастрофических потерях (50–100%) алгоритм делает большой шаг назад, чтобы быстро выйти из области значений нагрузки l, много превышающих отыскиваемое значение l_{max}. При незначительном же превышении потерями назначенного порогового значения Δ_ρ алгоритм осуществляет поиск в малой области вокруг текущего значения l.

Таким образом, задаваемое на n-ом шаге алгоритма значение текущей нагрузки $l^{(n)}$ определяется величиной разности $\left(\rho_n - \Delta_\rho\right)$, отложенной в процентах:

$$l^{(n+1)} = l^{(n)} + \alpha \cdot \left(l_{high}^{(n)} - l_{low}^{(n)}\right),$$

где $\alpha = \beta_0 + \beta_1 \cdot \min\left\{\frac{\rho_n - \Delta_\rho}{100\%}, 1\right\}$, значения β_0 и β_1 следует подбирать под конкретные условия задачи. Скорость сходимости данного метода зависит прежде всего от величины выбранных констант β_0 и β_1 в зависимости $\alpha(\rho_n)$. Адаптивный характер алгоритма обусловлен изменением величины шага $\Delta l = l^{(n+1)} - l^{(n)}$ величины нагрузки на тестируемых ПАК в зависимости от текущей величины превышения допустимых потерь, что концептуально близко к идее адаптивного взвешивания потерь в подходе [8], в котором осуществляется повышение сходимости стохастического градиентного спуска на основе адаптивной регулировки весов по значениям функции потерь.

Ниже при представлении псевдокода алгоритмов указаны только те дополнительные входные данные и промежуточные переменные, которые не содержались в описании приведенного выше Алгоритма 1. Введенные ранее обозначения повторно не поясняются.

Входные данные Алгоритма 2:

β_0 и β_1 – значения коэффициентов адаптивного поиска;

Псевдокод Алгоритма 2.

1) Присвоить n значение, равное 1.
2) Вычислить $l^{(n)} = \frac{1}{2} \cdot \left(l_{low}^{(n-1)} + l_{high}^{(n-1)}\right)$.
3) Получить результат измерения значения ρ_n.
4) Если $\rho_n \le \Delta_\rho$, то присвоить $l_{low}^{(n)}$ значение $l^{(n)}$, а $l^{(n+1)}$ – значение $\min\left\{l^{(n)} - \beta_1 \cdot \left(l_{high}^{(n-1)} - l^{(n-1)}\right),\ l_{high}^{(n-1)}\right\}$. Иначе присвоить $l_{high}^{(n)}$ значение $l^{(n)}$, а $l^{(\tau+1)}$ – значение $l^{(n)} + \alpha \cdot \left(l_{high}^{(n)} - l_{low}^{(n)}\right)$, где $\alpha = \beta_0 + \beta_1 \cdot \min\left\{\frac{\rho_n - \Delta_\rho}{100\%}, 1\right\}$.
5) Если $l_{high}^{(n)} - l_{low}^{(n)} \ge \Delta_l$, то присвоить n значение (n+1) и перейти к шагу 2. В противном случае выйти из процедуры, назначив $l_{\max} = \frac{1}{2} \cdot \left(l_{low}^{(n)} + l_{high}^{(n)}\right)$.

4.2. *Поиск с учетом предшествующих данных*

Идея о возможности использования ранее полученных результатов измерения значения l_{max} (исторические данные о производительности) для ускорения тестирования ПАК прямо или косвенно

отмечается в работах, обсуждающих ограничения детерминированных алгоритмов поиска в недетерминированных сетевых средах [7].

Алгоритм использует историческое значение $\hat{l}_{\max}$ пропускной способности $l_{\max}$, определенное для предыдущей версии ПО тестируемого ПАК. Поиск следует начинать, опираясь на границы интервала $\left(\hat{l}_{\max} \pm \hat{\Delta}_l\right)$, где $\hat{\Delta}_l$ – предел погрешности определения значения $\hat{l}_{\max}$. Это позволяет быстро либо подтвердить отсутствие значимых изменений ПС, либо определить направление изменений значения $l_{\max}$. Вначале следует провести измерение потерь ρ при нагрузке $l = \hat{l}_{\max}$. Если потери в норме, то далее следует выполнить дополнительное измерение при $l = \hat{l}_{\max} + \hat{\Delta}_l$. Если же потери окажутся превосходящими установленный для них порог, то следующее измерение следует выполнить при $l = \hat{l}_{\max} - \hat{\Delta}_l$. Если результат второго измерения окажется успешным (в смысле малости потерь), то значение $l_{\max}$ следует искать внутри получившегося промежутка возможных значений: в первом случае – $\left[\hat{l}_{\max} + \hat{\Delta}_l, l_{high}^{(0)}\right]$, во втором случае – $\left[\hat{l}_{\max} - \hat{\Delta}_l, \hat{l}_{\max}\right]$. Если результат второго измерения окажется сопряжен с недопустимыми потерями, то значение $l_{\max}$ следует искать: в первом случае – внутри интервала $\left[\hat{l}_{\max}, \hat{l}_{\max} + \hat{\Delta}_l\right]$, во втором – внутри промежутка $\left[l_{low}^{(0)}, \hat{l}_{\max} - \hat{\Delta}_l\right]$. Дальнейшие шаги следует осуществлять методом бинарного поиска. Если в рамках рассматриваемой процедуры требуется только подтвердить, что максимальная ПС не стала меньше по сравнению с предшествующей аттестацией, то в случае успешности первого из описанных измерений можно вернуть пользователю ответ в форме $l_{\max} \geq \hat{l}_{\max}$.

Псевдокод Алгоритма 3.

1) Присвоить значению нагрузки $l^{(0)}$ значение $\hat{l}_{\max}$. Получить результат измерения значения ρ_0 при значении нагрузки $l^{(0)}$.
2) Если $\rho_0 \leq \Delta_\rho$, то присвоить $l^{(1)}$ значение, равное $\hat{l}_{\max} + \hat{\Delta}_l$. Если нет – присвоить $l^{(1)}$ значение, равное $\hat{l}_{\max} - \hat{\Delta}_l$. Получить результат измерения значения ρ_1 при значении нагрузки $l^{(1)}$.
3) Если $\rho_1 \leq \Delta_\rho$ и $\rho_0 \leq \Delta_\rho$, то далее осуществлять бинарный поиск на интервале $\left[\hat{l}_{\max} + \hat{\Delta}_l, l_{high}^{(0)}\right]$. Если $\rho_1 \leq \Delta_\rho$ и $\rho_0 > \Delta_\rho$ – то на интервале $\left[\hat{l}_{\max}, \hat{l}_{\max} + \hat{\Delta}_l\right]$. Если $\rho_1 > \Delta_\rho$ и $\rho_0 > \Delta_\rho$ – то на интервале $\left[l_{low}^{(0)}, \hat{l}_{\max} - \hat{\Delta}_l\right]$. Если $\rho_1 > \Delta_\rho$ и $\rho_0 \leq \Delta_\rho$ – то на интервале $\left[\hat{l}_{\max} - \hat{\Delta}_l, \hat{l}_{\max}\right]$.

4.3. *Интерполяционный поиск* [9]

Пусть в результате первого измерения при определении максимальной ПС полученное значение потерь не превзошло установленных для него потерь, а второе измерение показало большие потери, чем допустимо, что является обычной ситуацией, если в качестве значений нагрузки выбираются соответственно значения $l_{low}^{(0)}$ и $l_{high}^{(0)}$. В предположении монотонного изменения уровня потерь с ростом объема пропускаемого трафика следующее значение нагрузки в данном методе поиска предполагается выбирать с помощью линейной интерполяции, используемой для предсказания нагрузки, при которой потери станут равны пороговому значению Δ_ρ =0,1%. После проверки уровня потерь в получившейся точке следует сузить интервал поиска значения нагрузки $l_{\max}$ и повторить процедуру.

Промежуточные переменные Алгоритма 4:

$\rho_{middle}^{(n)}, \rho_{high}^{(n)}$ – результаты измерения уровня потерь при в середине и в верхней границе интервала локализации такой нагрузки l_{max}, при которой обеспечивается максимальная пропускная способность, определенные на n-й итерации алгоритма поиска.

Псевдокод Алгоритма 4.

1) Присвоить n значение, равное 1. Выполнить измерение уровня потерь $\rho_{high}^{(n-1)}$ при нагрузке, равной $l_{high}^{(n-1)}$.

2) Выполнять, пока выполнено $\frac{1}{2} \cdot \left(l_{high}^{(n-1)} - l_{low}^{(n-1)}\right) \geq \Delta_l$.

2.1) Получить результат измерения $\rho_{middle}^{(n-1)}$ уровня потерь при нагрузке, равной $l_{middle}^{(n-1)} = \frac{1}{2} \cdot \left(l_{low}^{(n-1)} + l_{high}^{(n-1)}\right)$.

2.2) Вычислить линейное предсказание значения $l_{\max}$ как $l^{(n)} = l_{middle}^{(n-1)} + \frac{\Delta_\rho - \rho_{middle}^{(n-1)}}{\rho_{high}^{(n-1)} - \rho_{middle}^{(n-1)}} \cdot \left(l_{high}^{(n-1)} - l_{middle}^{(n-1)}\right)$.

2.3) Получить результат измерения значения ρ_n уровня потерь при значении нагрузки $l^{(n)}$.

2.4) Если $\rho_n < \Delta_\rho$, то присвоить $l_{low}^{(n)}$ значение $l^{(n)}$. В противном случае присвоить $l_{high}^{(n)}$ значение $l^{(n)}$, а $\rho_{high}^{(n)}$ – значение ρ_n.

2.5) Увеличить значение n на единицу.

3) Вернуть $l_{\max} = \frac{1}{2} \cdot \left(l_{low}^{(n-1)} + l_{high}^{(n-1)}\right)$.

4.4. *Регрессионный поиск с накоплением*

Данный подход предполагает использовать всю получаемую в ходе процедуры аттестации ПАК измерительную информацию для лучшей оценки предполагаемого значения $l_{\max}$ с помощью линейной регрессии. На каждом шаге по всем полученным точкам $(l^{(i)}, \rho_i)$, $i = 1, 2, \ldots, n$ строится регрессионная модель, которая предсказывает положение значение $l_{\max}$. Такой подход устойчив к шуму и использует всю доступную информацию.

Псевдокод Алгоритма 5.

1) Получить результаты измерения ρ_1 и ρ_2 уровня потерь при значениях нагрузки $l^{(1)} = \frac{1}{2} \cdot \left(l_{low}^{(0)} + l_{high}^{(0)}\right)$ и при $l^{(2)} = l_{high}^{(0)}$ соответственно. Присвоить n значение, равное 2.
2) Выполнять, пока выполнено условие $\frac{1}{2} \cdot \left(l_{high}^{(n-2)} - l_{low}^{(n-2)}\right) \geq \Delta_l$.

2.1) Вычислить коэффициенты линейной регрессии по результатам всех выполненных измерений: $a = \frac{n \cdot \sum_{i=1}^{n} l^{(i)} \cdot \rho_i - \left(\sum_{i=1}^{n} l^{(i)}\right) \cdot \left(\sum_{i=1}^{n} \rho_i\right)}{n \cdot \left(\sum_{i=1}^{n} \left(l^{(i)}\right)^2\right) - \left(\sum_{i=1}^{n} l^{(i)}\right)^2}$ и $b = \frac{1}{n} \cdot \sum_{i=1}^{n} \rho_i - \frac{a}{n} \cdot \sum_{i=1}^{n} l^{(i)}$.

2.2) Вычислить $l^{(n+1)} = \frac{\Delta_\rho - b}{a}$. Получить результат измерения ρ_{n+1} при данном значении нагрузки.

2.3) Если $\rho_{n+1} < \Delta_\rho$ то присвоить $l_{low}^{(n-1)}$ значение $\max\left\{l_{low}^{(n-2)}, l^{(n+1)}\right\}$. В противном случае присвоить $l_{high}^{(n-1)}$ значение $\min\left\{l_{high}^{(n-2)}, l^{(n+1)}\right\}$.

2.4) Увеличить значение n на единицу.

3) Вернуть $l_{\max} = \frac{1}{2} \cdot \left(l_{low}^{(n-2)} + l_{high}^{(n-2)}\right)$.

Данный подход использует все результаты ранее выполненных измерений (включая те, что указывали на превышение порога допустимых потерь) и в этом отношении задействует всю имеющуюся информацию.

4.5. *Поиск с адаптивным шагом по крутизне*

Данный подход основан на оценке значений $s(l) = d\rho(l)/dl$ производной функции $\rho(l)$ значений потерь от нагрузки (крутизны зависимости) и представляет собой вариант классического метода Ньютона для поиска. Величина шага по нагрузке обратно пропорциональна значению производной $d\rho(l)/dl$: чем оно больше, тем меньше шаг для уточнения границ области поиска $l_{\max}$. На основе результатов последних по времени выполненных измерений потерь ρ_n, ρ_{n-1}, … при значениях соответствующих нагрузок $l^{(n)}$, $l^{(n-1)}$, … тем или иным способом оценивают значение производной s – например, с помощью первой конечной разности:

$$s \approx \frac{\rho_n - \rho_{n-1}}{l^{(n)} - l^{(n-1)}}.$$

Затем шаг Δl изменения нагрузки определяют как $\Delta l = \left(l_{high}^{(n)} - l_{low}^{(n)}\right) \cdot \min\{1, k/|s|\}$, где k – заданный параметр регулировки.

Дополнительные входные данные Алгоритма 6:

$k>0$ – заданное значение параметра регулировки.

Дополнительные промежуточные переменные Алгоритма 5:

s – оценка производной (крутизны) функции зависимости потерь $\rho(l)$ от величины нагрузки l.

Псевдокод Алгоритма 6.

1) Присвоить n значение, равное 1.

2) Проверить, выполнено ли неравенство $l_{high}^{(n-1)} - l_{low}^{(n-1)} \geq \Delta_l$. Если да, то перейти к п. 3. Если нет, то перейти к п. 9.

3) Вычислить $l^{(n)} = \frac{1}{2} \cdot \left(l_{low}^{(n-1)} + l_{high}^{(n-1)}\right)$.

4) Получить результат измерения значения ρ_n.

5) Если выполнено условие $\rho_n \leq \Delta_\rho$, то присвоить $l_{low}^{(n)}$ значение, равное $l^{(n)}$, в противном случае – присвоить $l_{high}^{(n)}$ значение $l^{(n)}$.

6) Если $n<2$, то перейти к п. 8. В противном случае вычислить значение $s = \frac{\rho_n - \rho_{n-1}}{l^{(n)} - l^{(n-1)}}$.

7) Если $s \neq 0$, то присвоить величине $l^{(n)}$ значение, равное $\left(l_{high}^{(n)} - l_{low}^{(n)}\right) \cdot \min\{1, k/|s|\}$. В противном случае присвоить $l^{(n)}$ значение $l^{(n)} = \frac{1}{2} \cdot \left(l_{low}^{(n)} + l_{high}^{(n)}\right)$.

8) Присвоить n значение, равное $n + 1$. Перейти к п. 2.

9) Вернуть $l_{\max} = \frac{1}{2} \cdot \left(l_{low}^{(n-1)} + l_{high}^{(n-1)}\right)$.

V. Сравнение рассмотренных подходов

Было выполнено имитационное моделирование для сравнения результатов, получаемых при применении рассмотренных методов поиска для измерения $l_{\max}$. Для этого была разработана математическая модель на языке Matlab со следующими параметрами и характеристиками:

- истинное значение $l_{\max}$ пропускной способности варьировалось от 800 до 1200 условных единиц;
- в качестве возможной функции потерь от нагрузки была использована линейная функция с наложенным случайным шумом, подчиняющимся нормальному распределению с нулевым математическим ожиданием и заданной дисперсией σ^2:

$$\rho(l) = \max\{0, \alpha \cdot (l - l_{\max}) + \varepsilon\},$$

где α – крутизна роста потерь после превышения порога, ε – случайный шум (были приняты значения $\alpha = 0{,}05$ и $\sigma = 1\%$);

- учтена возможность просадки производительности (т.е. значения $l_{\max}$) для анализируемой новой версии ПО по сравнению с предшествующей его версией, реализованная в форме уменьшения истинного значения $l_{\max}$ на (1÷5)%.

Для каждого из рассмотренных методов поиска было выполнено $N=10^3$ симуляций. Требуемая точность определения значения $l_{\max}$ составляла $\Delta_l = 10$ единиц, а предел Δ_ρ составлял 0,1. Для адаптивного поиска с весовой функцией потерь были взяты значения коэффициентов β_0=–0,5 и β_1=–1,0. В качестве основных метрик для сравнения результатов были взяты следующие: среднее количество итераций измерений m до сходимости; средняя абсолютная ошибка MAE оценки ПС.

Полученные результаты представлены на рис. 1 и в табл. 1. Рис. 1 наглядно показывает, что алгоритмы с элементами адаптивности (особенно регрессионный поиск) характеризуются меньшим разбросом в результатах и более низкими медианными значениями числа итераций, которые необходимо выполнить для получения значения максимальной ПС, по сравнению с классическим бинарным поиском. Табл. 1 содержит количественные результаты данного сравнения: в ней приведены значения среднего числа m потребовавшихся итераций и средней абсолютной ошибки (MAE) определения $l_{\max}$, оцененные по $N=10^3$ выполненных симуляций.

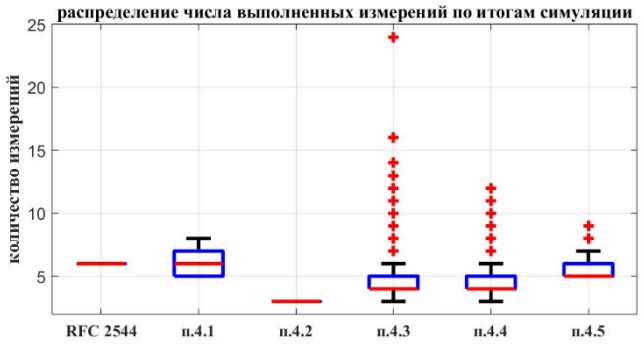

Рис. 1. Сравнение эффективности рассмотренных алгоритмов поиска значения максимальной пропускной способности

ТАБЛИЦА I. КОЛИЧЕСТВЕННЫЕ ПОКАЗАТЕЛИ ЭФФЕКТИВНОСТИ РАССМОТРЕННЫХ АЛГОРИТМОВ ПОИСКА ЗНАЧЕНИЯ МАКСИМАЛЬНОЙ ПРОПУСКНОЙ СПОСОБНОСТИ

Алгоритм	*m*	MAE
Бинарный поиск	6,0±0,1	2,20±0,05
Адаптивный поиск с весовой функцией потерь	6,1±0,1	2,45±0,05
Поиск с учетом предшествующих данных	3,0±0,1	3,40±0,01
Интерполяционный поиск	4,4±0,1	2,13±0,05
Регрессионный поиск с накоплением	4,5±0,1	2,33±0,04
Поиск с адаптивным шагом по крутизне	5,6±0,1	2,69±0,06

Наименьшее среднее число итераций достигнуто для поиска с учетом предшествующих данных (m=3,0) и интерполяционного поиска (m=4,4), что в (1,4÷2,0) раза меньше, чем у классического бинарного поиска (m=6,0). Достигнуто сокращение числа измерений на (27÷50)% при сохранении требуемой точности (MAE = 2,1–3,4 против 2,2 у бинарного поиска).

VI. ЗАКЛЮЧЕНИЕ

Разработанные алгоритмы позволяют сократить время тестирования ПО ПАК для обработки сетевого трафика в более чем в 1,4÷2,0 раза (от 27% до 50% от числа итераций) по сравнению со стандартной процедурой [6], применяемой в настоящее время. Улучшение достигнуто за счет использования результатов предшествующих измерений и аппроксимации функции потерь. Внедрение соответствующих методов поиска в конвейер непрерывной интеграции для разработки сетевого оборудования позволяет ускорить вывод новых версий ПО без потери качества.

СПИСОК ЛИТЕРАТУРЫ

[1] Тюменцев, Д.В. Автоматизация тестирования в DevOps: подходы и лучшие практики // Международный журнал гуманитарных и естественных наук. 2024. № 2-2(89). С. 156-159. – DOI: 10.24412/2500-1000-2024-2-2-156-159.

[2] Kolawole, I., Fakokunde, A. (2024). Improving Software Development with Continuous Integration and Deployment for Agile DevOps in Engineering Practices. International Journal of Computer Applications Technology and Research, 14(1), 25-39. DOI: 10.7753/ijcatr1401.1002.

[3] Bhanushali, A. (2023). Challenges and Solutions in Implementing Continuous Integration and Continuous Testing for Agile Quality Assurance. International Journal of Science and Research, 12(10), 1626–1644. DOI: 10.21275/sr231021114758.

[4] Marijan, D., Liaaen, M., Sen, S. (2018). DevOps Improvements for Reduced Cycle Times with Integrated Test Optimizations for Continuous Integration. In: 2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC) (pp. 22–27). DOI: 10.1109/compsac.2018.00012.

[5] Bradner, S. (1991). Benchmarking Terminology for Network Interconnection Devices. RFC Editor. Paper rfc1242. DOI: 10.17487/rfc1242.

[6] Bradner, S., McQuaid, J. (1999). Benchmarking Methodology for Network Interconnect Devices. RFC Editor. Paper rfc2544. DOI: 10.17487/rfc2544.

[7] Lencse, G., Kovács, Á., Shima, K. (2020). Gaming with the Throughput and the Latency Benchmarking Measurement Procedures of RFC 2544. International Journal of Advances in Telecommunications, Electrotechnics, Signals and Systems, 9(2), 10. DOI: 10.11601/ijates.v9i2.288.

[8] Haimovich, D., Karamshuk, D., Linder, F., Tax, N., Vojnovic, M. (2024). On the Convergence of Loss and Uncertainty-based Active Learning Algorithms. Advances in Neural Information Processing Systems, 37, 122770-122810. DOI: 10.48550/arXiv.2312.13927.

[9] Multiple Loss Ratio Search: draft-ietf-bmwg-mlrsearch-15 (2020). [Электронный ресурс]. Формат доступа: свободный. Дата обращения: 01.02.2026. URL: https://datatracker.ietf.org/doc/draft-ietf-bmwg-mlrsearch/