УДК 004.04

Д. В. Шиман, доц., канд. техн. наук;
Е. С. Рябченко, магистрант
(БГТУ, г. Минск)

МЕТОД СРАВНЕНИЯ ЭФФЕКТИВНОСТИ АЛГОРИТМОВ, РЕАЛИЗОВАННЫХ НА JAVASCRIPT И WEBASSEMBLY

В настоящее время существует огромное множество языков программирования. Каждый из них создан для решения определенных задач. Но, порой возникает ситуация, в которой целевая аудитория существующего программного средства растет, а ее потребности изменяются. Разработчикам приходится адаптировать программное средство под новые требования. Иногда, это может быть затратно как по времени, так и по средствам.

Десктопные приложения необходимо скачивать и инсталлировать. А web-приложения не требуют от пользователя этого, достаточно просто ввести адрес нужного сайта. Но, что делать разработчикам, если существует web-приложение с большим количеством клиентов, а его производительность недостаточна из-за сложных алгоритмов, которые не могут быть выполнены быстрее с использованием JavaScript?

В качестве цели для данной работы был выбран поиск способа ускорения существующего веб-приложения.

Для достижения поставленной цели необходимо решить следующие задачи:

1. Проанализировать существующие подходы;
2. Выбрать наиболее перспективный подход;
3. Выбрать алгоритмы, которые необходимо испытать;
4. Реализовать выбранные алгоритмы;
5. Сравнить результаты.

Существуют два подхода, которые могут быть использованы для ускорения выполнения JavaScript: asm.js и WebAssembly.

asm.js — это подмноженство JS, имеющее более высокую оптимизацию за счет статических типов, которые во время выполнения дают бóльшую производительность.

WebAssembly — это бинарный формат инструкций для виртуальной машины, результат компиляции с языка высокого уровня. Поскольку это бинарный формат, он не годится для чтения человеком, зато является наиболее быстрым на этапе выполнения в сравнении с asm.js, поскольку более приближен к машинному языку. Таким образом, был выбран WebAssembly.

Важно понимать, что WebAssembly — это не язык программирования, а формат данных, и он никак не связан с вебом напрямую, и может быть использован где угодно. WebAssembly может быть получен из многих языков высокого уровня, что позволяет портировать библиотеки из, например, C++.

JavaScript и WASM работают в одном и том же контексте, и исполняются одним и тем же движком. Из WebAssembly можно вызывать JavaScript. Это может быть вызов функции с передачей аргументов и возвратом значения, либо это может быть просто выполнение произвольной строки как JavaScript -кода.

Для проверки быстродействия необходимо выбрать алгоритмы, скорость которых будет измерена. В моем случае выбор был сделан в пользу следующих алгоритмов: MD5, SHA256, SHA512.

Для тестирования времени работы алгоритмов было разработано приложение, позволяющее провести заданное количество испытаний, сгенерировать случайную строку заданной длины, измерить время выполнения алгоритма. Было выбрано 50 длин строк, а для каждой длины по 100 испытаний. Суммарно было проведено 15000 испытаний для трех алгоритмов.

Для MD5 нельзя однозначно сказать, что выигрывает JavaScript или WebAssembly, поскольку на разных длинах строк получены разные результаты.

Для алгоритмов SHA256 и SHA512 ситуация оказалась куда более однозначная: для SHA256 в более чем 50% испытаний победителем оказывался WebAssembly, а согласно усредненным данным, он и вовсе должен быть использован практически во всех случаях, а для SHA512 и более 99% испытаний оказывались выигрышными для WebAssembly.

На основе проведенных испытаний был разработан алгоритм выбора наилучшего пути решения. Он представляет собой функцию, которая принимает 2 параметра: необходимый алгоритм и значение, которое необходимо передать. Данная функция использует результаты, полученные в ходе испытаний, описанных ранее.