Statystyczna ocena wydajności serwera symulatora mieszanej rzeczywistości

Piotr Golański, Michał Golański
Instytut Techniczny Wojsk Lotniczych, ul. Ks. Bolesława 6, 01-494 Warszawa

Streszczenie: Mimo ciągłego wzrostu mocy obliczeniowych współczesnych komputerów, ich możliwości w wielu dziedzinach są niewystarczające. Przykładem może być symulacja procesów fizycznych, stosowana we wszelkiego rodzaju systemach szkoleniowych opartych na technologii wirtualnej lub mieszanej rzeczywistości. Wynika to z rosnących wymagań stawianych tym systemom na osiąganie coraz większych dokładności w odwzorowaniu rzeczywistości. Rozwiązaniem tego problemu jest wykorzystanie symulacji rozproszonej, która dzięki zastosowaniu rozwiązań chmurowych umożliwia zdalny dostęp do mocy obliczeniowych specjalizowanych serwerów, tym samym zapewnia zdalne prowadzenie szkolenia. Krytycznym zagadnieniem staje się zapewnienie efektywnej komunikacji zapewniającej wymianę danych między klientami a serwerem symulacji w czasie rzeczywistym. W artykule przedstawiono propozycję budowy serwera oraz protokołu komunikacji do zastosowania w chmurowej wersji symulatora śmigłowca Mi-17 opartego na technologii mieszanej rzeczywistości. Zaproponowano metodę pomiaru prędkości łącza opartego na tym protokole oraz dokonano statystycznego opisu uzyskanych wyników.

Słowa kluczowe: serwer chmurowy, symulacja rozproszona, mieszana rzeczywistość, dopasowanie rozkładów, testy zgodności

1. Wprowadzenie

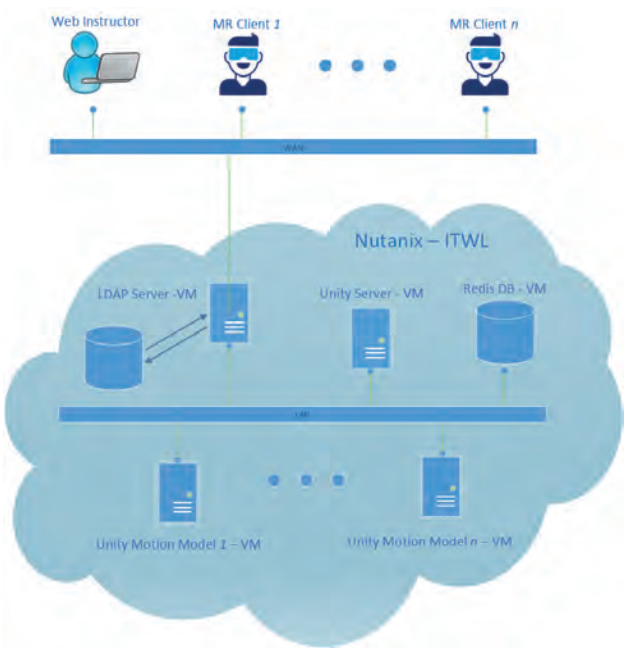
Od wielu lat technologia chmurowa znajduje zastosowanie w nauce, technice i przemyśle. Pozwala udostępniać zasoby komputerowe, takie jak serwery, pamięć masowa, bazy danych a także wyniki działania aplikacji zdalnym klientom [8]. Dzięki temu możliwe jest m.in. wyodrębnienie krytycznych algorytmów, np. z punktu widzenia ich poufności, umieszczenia ich na serwerze i udostępnianie jedynie wyników ich działania jako odpowiedzi na zapytania zdalnych klientów. Takie założenie zostało przyjęte przy budowie symulatora chmurowego statku powietrznego realizowanego w Instytucie Technicznym Wojsk Lotniczych (rys. 1).

Środowisko chmurowe dla symulatora ma stanowić hiperkonwergentna platforma Nutanix, uruchomiona w ramach Krajowego Ośrodka Badawczego Inteligentnych Materiałów Kompozytowych osadzoną we własnej infrastrukturze Instytutu Technicznego Wojsk Lotniczych. Środowisko tworzą dwa główne elementy funkcjonalne: w warstwie sprzętowej – klaster Nutanix i sprzęt firmy Fujitsu, w warstwie programowej – pakiet obliczeniowy ANSYS.

Autor korespondujący:
Piotr Golański, piotr.golanski@itwl.pl

Artykuł recenzowany
nadesłany 26.11.2024 r., przyjęty do druku 11.07.2025 r.

Rozproszona symulacja mieszanej rzeczywistości MR będzie odbywać się równocześnie na wielu stanowiskach klienckich MR Client i, z zainstalowanym oprogramowaniem symulacyjnym mieszanej rzeczywistości (będzie to symulator MR



Rys. 1. Schemat koncepcyjny wielostanowiskowej symulacji z użyciem środowiska chmurowego Nutanix
Fig. 1. Concept scheme of multi-station simulation using the Nutanix cloud environment

śmigłowca Mi-17 [7]), wyposażonych w rzeczywiste organy sterowania, na których będą zaimplementowane graficzne modele ich cyfrowych bliźniaków oraz wirtualne modele graficzne obiektów technicznych wraz z otoczeniem. Stanowiska klienckie MR Client *i* będą stanowić węzły symulacji 3D. Dla każdego stanowiska po zalogowaniu się w sieci rozległej do chmury Nutanix, będzie przydzielana maszyna wirtualna Unity Motion Model *i*, z modelem matematycznym ruchu obiektu, stanowiąc lokalny węzły symulacji. Komunikacja między wszystkimi węzłami symulacyjnymi będzie się odbywać z wykorzystaniem Serwera Unity.

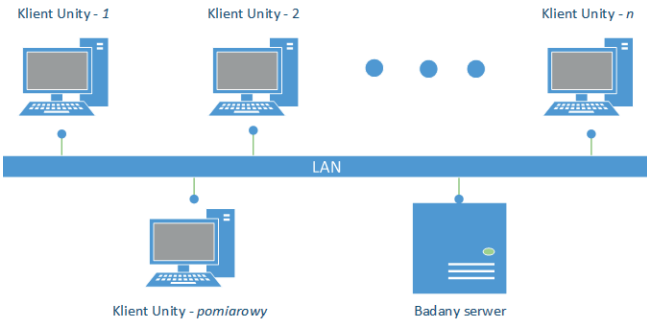
Krytycznym zagadnieniem będzie tu zagwarantowanie efektywnej komunikacji umożliwiającej wymianę danych między zdalnymi stanowiskami symulatora a serwerem w czasie rzeczywistym. Czasy odpowiedzi serwera nie stanowią większego problemu, jeśli obiekty generowane na różnych węzłach MR Client *i* są od siebie oddalone w symulowanej przestrzeni. Problem pojawia się, gdy symulowane obiekty znajdują się w bliskiej odległości, np. pierwszy i drugi pilot w kabinie lecącego śmigłowca. W takich sytuacjach bardzo ważne jest zapewnienie krótkich czasów odpowiedzi serwera.

2. Sformułowanie problemu

W celu przeprowadzenia badań wydajnościowych serwera, zbudowano w warunkach laboratoryjnej sieci lokalnej zestaw dwóch stanowisk symulacyjnych. Na jednym stanowisku symulowany był ruch uproszczonej platformy z obiektem 3D, symulującym pierwszego pilota, na drugim symulowano model 3D drugiego pilota, który we wspólnej przestrzeni znajdował się na tej samej ruchomej platformie. Komunikacja między stanowiskami odbywała się z wykorzystaniem oprogramowania Unity Server VM. Oprogramowanie zaimplementowano w oparciu o technologię kontenerową, wykorzystując platformę *Docker*. Kontenery pobrano z rejestru dockerhub [18] jako dwa obrazy *Redis* (baza danych) i *Alpine Linux* (Serwer Unity). Na bazie *Alpine* zainstalowano właściwy serwer Unity Server. W protokole komunikacyjnym zastosowano mechanizmy *WebSocket* [11] oraz serializację *JSON* [16]. Ta architektura w dalszej części artykułu oznaczono jako A1 (rys. 2).

W trakcie badań, z użyciem tak zaimplementowanego serwera, zaobserwowano utratę pakietów i niesatysfakcjonującą wydajność. Dlatego zdecydowano zoptymalizować jego budowę, tak aby wyeliminować powyższe wady. Przede wszystkim uproszczono serwer przez zaimplementowanie bazy danych wewnątrz serwera jako zasobu pamięci z synchronizowanym dostępem. Zmieniono serializację *JSON* na *MsgPack* [17] z opcją serializacji bazującej na tablicach (ang. *array-based serialization* [20]). W dalszej części pracy tę architekturę oznaczono jako A2. Przeprowadzono następnie pomiary porównawcze czasów opóźnień dla obydwu serwerów.

Badania obydwu architektur przeprowadzono na zestawie składającym się z jednego komputera z systemem Linux Ubuntu w wersji 22.04 i uruchomionym na nim oprogramowaniem badanego serwera w dwóch wersjach oraz z sześciu stanowisk Klient Unity z systemem Windows 10/11 i oprogramowaniem symulacyjnym (rys. 3). Oprogramowanie na każdym stanowisku zawierało scenę Unity ze wszystkimi symulowanymi obiektami 3D, przy czym symulacja ruchu każdego z nich odbywała się na innym stanowisku. Synchronizacja danych odbywała się z częstotliwością odświeżania obrazu Unity (60 fps), przez wysłanie żądań klientów do serwera z podaniem wektora pozycji i rotacji. W odpowiedzi na żądanie serwer wysyłał do klientów wektory pozycji i rotacji obiektów symulowanych na pozostałych klientach. Interfejs na stanowiskach Klient Unity pozwalał jedynie na obserwację sceny.



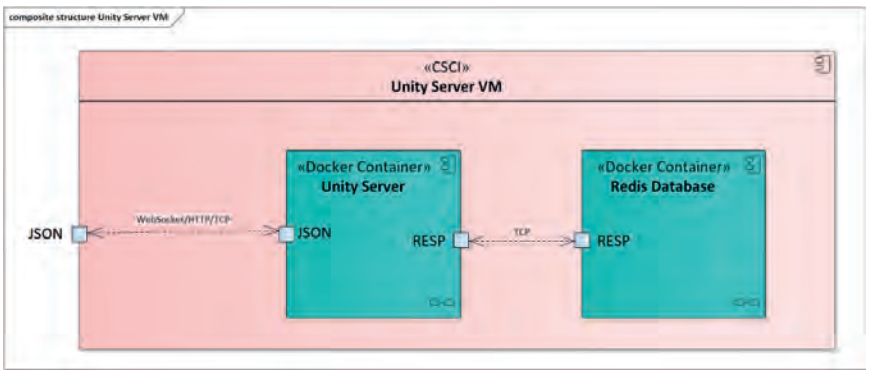
Rys. 3. Układ do pomiaru czasów odpowiedzi serwera
Fig. 3. System for measuring server response times

Do pomiaru czasów opóźnień zastosowano oprogramowanie *WireShark* zainstalowane na jednym z klientów, określonym jako Klient Unity – *pomiarowy*. Pomiary polegały na przechwytywaniu pakietów przez aplikację *WireShark* w trakcie pracy klientów. Po zebraniu danych (w prezentowanej pracy były to dane obejmujące czas około 1 minuty), proces akwizycji był zatrzymywany, a wyniki eksportowane i zapisywane w plikach z rozszerzeniem *pcapng*. W celu uzyskania czasów opóźnień z danych zawartych w plikach *pcapng*, został opracowany skrypt w języku *Python*, który generował pliki tekstowe z zarejestrowanymi czasami opóźnień.

3. Analiza statystyczna czasów odpowiedzi serwerów

3.1. Przegląd metod

Do analizy zebranych danych pomiarowych, które są związane z czasowym aspektem zachowania się systemów, stosuje się techniki zdominowane przez deterministyczną analizę czasu odpowiedzi RTA (ang. *Response-Time Analysis*) [13], które opierają się na restrykcyjnych założeniach dotyczących zachowania sys-



Rys. 2. Diagram architektury A1
Fig. 2. The A1 architecture diagram

temu i zwracają wartość bezwzględną czasu reakcji dla najgorszego przypadku WCRT (ang. *Worst-Case Response Time*). Bardzo często te wartości są zbyt duże w porównaniu z rzeczywistymi wynikami.

W celu usunięcia tych wad, dla systemów, których nie da się poprawnie przeanalizować za pomocą deterministycznego RTA, proponowane jest podejście statystyczne oparte na teorii wartości ekstremalnych EVT (ang. *Extreme Value Theory*), która rozszerza centralne twierdzenie graniczne na ogony rozkładu i jest stosowana do modelowania zdarzeń rzadkich [9]. W pracy do analizy statystycznej wykorzystano rozkład uogólnionych wartości maksymalnych GEV (ang. *Generalized Extreme Value)*, który łączy trzy rozkłady: Gumbela, Frecheta i Weibula. Ponadto zastosowano metodę estymacji parametrów statystycznych czasów opóźnień [12], składającą się z trzech kroków:

1. wstępna identyfikacja rozkładów kandydujących,
2. estymacja parametrów rozkładu,
3. weryfikacji jako ocena dobroci i dopasowania.

Do wstępnej identyfikacji wybrano rozkłady badane w pracy [15]. Tak więc do dalszych rozważań przyjęto następujące rozkłady:

- normalny,
- logarytmiczno-normalny,
- Pareto,
- uogólniony rozkład wartości maksymalnych GEV (oryginalnie w pracy [15] GEV jest zastąpiony rozkładem Weibula).

Estymację parametrów wybranych rozkładów przeprowadzono metodą największej wiarygodności MLE (ang. *Maximum Likelihood Estimation*), w której stosuje się funkcję wiarygodności, określoną dla rozpatrywanego w pracy przypadku w postaci:

$$L(\theta) = \sum_{i=1}^{n} f(t_i \mid \theta) \tag{1}$$

gdzie: t_i – i-ty z n zaobserwowanych czasów odpowiedzi; $f(\cdot \mid \theta)$ – gęstość prawdopodobieństwa dla sparametryzowanego rozkładu.

Następnie poszukiwane są wartości parametrów $\theta = [\theta_1 \quad \theta_2 \quad \dots \quad \theta_m]$ w taki sposób, aby zmaksymalizować prawdopodobieństwo wygenerowania obserwacji dla oszacowanych parametrów [14]. Poszukiwane parametry są rozwiązaniem układu równań [6]:

$$\frac{\partial \log L}{\partial \theta_i} = 0 \tag{2}$$

W praktycznych rozwiązaniach stosowane są dedykowane narzędzia, np. *fitdistrplus* [5] albo bardziej uniwersalne środowiska obliczeniowe, jak Statistica lub MATLAB, który został użyty w niniejszej pracy.

Do oceny zgodności dopasowania zastosowano test Kołmogorowa-Smirnowa, podobnie jak zaproponowano w [4]. W tym teście miarą zgodności jest statystyka [6]:

$$D_n = \sup_{-\infty \le x \le +\infty} \left| F(\cdot \mid \theta) - S_n(t) \right| \tag{3}$$

gdzie: $F(\cdot \mid \theta)$ – dystrybuanta dla rozkładu o parametrach wyznaczonych z równania (2); $S_n(t)$ – dystrybuanta empiryczna w próbie prostej z n elementów.

W teście przyjmuje się, że hipoteza o zgodności rozkładu empirycznego i teoretycznego będzie odrzucona na poziomie istotności α, jeśli spełnione będzie:

$$P\left(D_n > D_\alpha\right) = \alpha \tag{4}$$

gdzie: D_α – wartość krytyczna dla danego poziomu istotności.

Prawdopodobieństwo P, zwane dalej *wartością p*, określa istotność statystyczną, zwiększającą rygor wniosków wysnutych z danych [2].

3.2. Wyniki

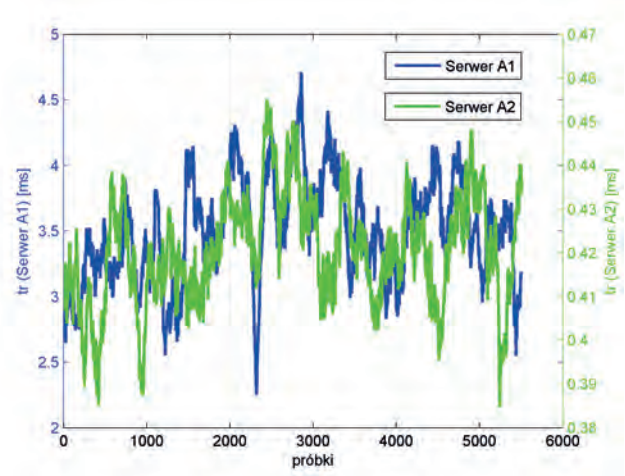
Na wstępie analizy czasów odpowiedzi obydwu serwerów należy zaznaczyć, że już samo zastąpienie serializacji *JSON* przez opartą na tablicach serializacją *MsgPack*, spowodowało zmniejszenie obciążenia sieci, przez zmniejszenie wielkości przesyłanych pakietów. W tabeli 1 zestawiono wartości charakteryzujące rozmiary parametrów dla obydwu rodzajów architektury.

Tab. 1. Parametry rozkładów rozmiaru pakietów w bajtach
Tab. 1. Distribution parameters for packet sizes in bytes

	Dominanta	Mediana	Średnia
A1	98	814	675
A2	182	182	194

Z danych zamieszczonych w tabeli wynika, że rozkłady wielkości pakietów są lewoskośne – z bardzo dużym rozrzutem wartości ich parametrów statystycznych dla architektury A1.

Dla zwiększenia czytelności i porównania czasów odpowiedzi przeprowadzono estymację wartości średnich czasu odpowiedzi stosując ruchome okno uśredniające obejmujące 100 próbek. W wyniku uzyskano następujące dwa przebiegi oznaczone jako Serwer A1 i Serwer A2. Na rys. 5 przedstawiono uzyskane wyniki czasów odpowiedzi.



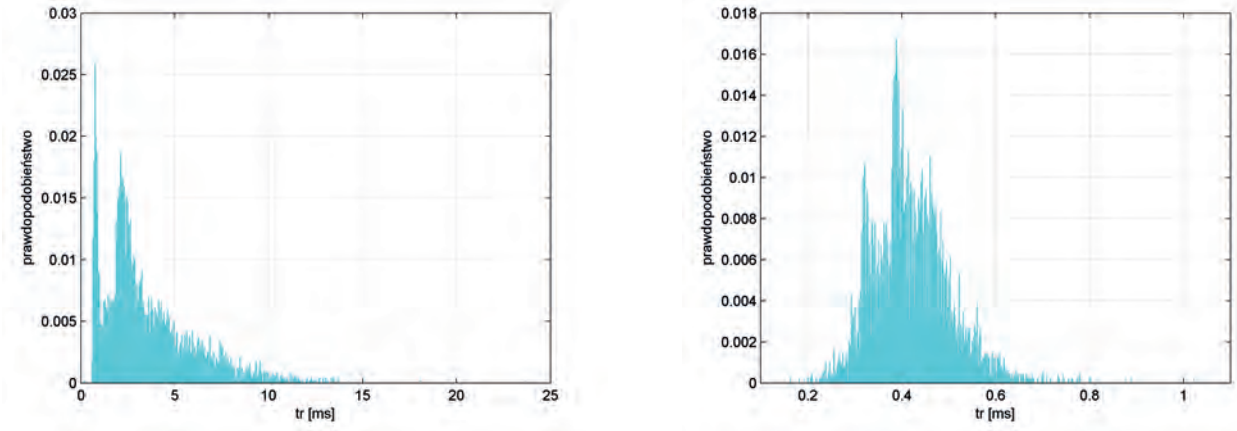
Rys. 5. Histogram czasu odpowiedzi dla obu rodzajów architektury
Fig. 5. Histogram of response times for both architectures

Pobieżna analiza pozwala stwierdzić, że czasy odpowiedzi w przypadku serwera A2 są o rząd wielkości mniejsze niż w przypadku serwera A1. Dla przedstawionych danych przeprowadzono nieparametryczne testy trendu (ang. *Reverse Arrangement Test*) [1], które potwierdziły stacjonarność pierwszego rzędu.

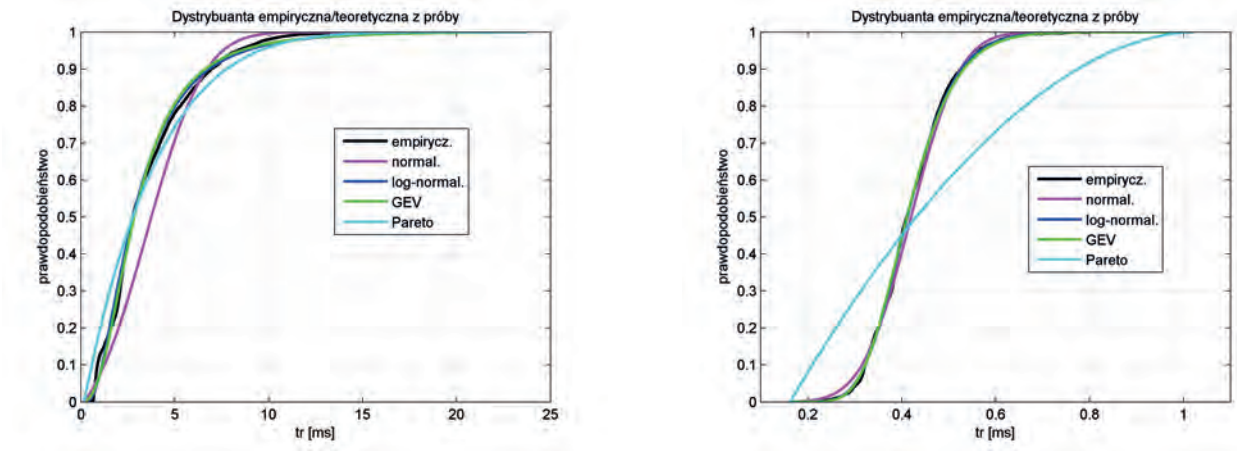
Na kolejnym rysunku przedstawiono histogramy dla obydwu czasów odpowiedzi. W przypadku serwera A1 można zaobserwować dwie wartości modalne dla czasów 0,75 ms i 2,17 ms, natomiast dla drugiego serwera są to wartości 0,32 ms, 0,39 ms i grzebieniowy odcinek dla czasów 0,42 ms, 0,44 ms i 0,46 ms.

W wyniku przeprowadzonego dopasowania rozkładów uzyskano wyniki, które przedstawiono w postaci zestawienia dystrybuanty empirycznej z rozkładami teoretycznymi, uzyskanymi metodą największej wiarygodności.

Z analizy wykresów (rys. 7) widać, że dla serwera A1 występują duże różnice dopasowania dla rozkładu normalnego, nato-



Rys. 6. Histogram czasów odpowiedzi dla serwera A1 (z lewej) i A2 (z prawej)
Fig. 6. Histogram of response times for server A1 (left) and A2 (right)



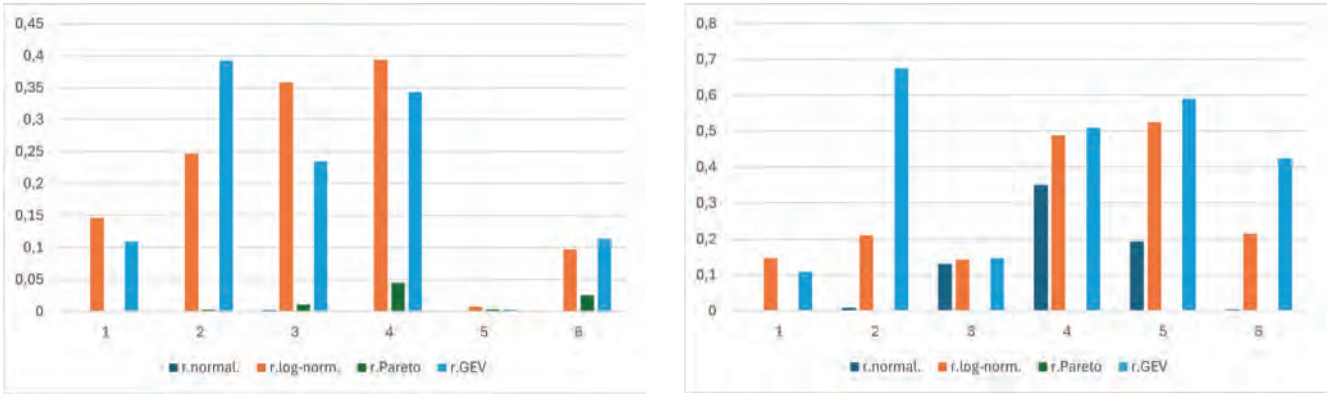
Rys. 7. Dystrybuanty empiryczne i teoretyczne czasów odpowiedzi dla serwera A1 (z lewej) i A2 (z prawej)
Fig. 7. Empirical and theoretical distributions of response times for server A1 (left) and A2 (right)

miast dla serwera A2 występuje bardzo duże niedopasowanie dla rozkładu Pareto. Mimo lepszego dopasowania pozostałych rozkładów, żaden z nich nie przeszedł testów zgodności Kołmogorowa-Smirnowa. Dokładniejsza analiza wykazała, że zmienność wartości średniej czasów odpowiedzi w trakcie pomiaru zmieniała się na tyle, że dla poprawnego dopasowania należało estymować rozkłady w krótszych przedziałach, podobnie jak dla procesów lokalnie stacjonarnych [3]. W związku z powyższym dla otrzymanych wyników zbudowano sześć rozłącznych przedziałów o liczności $n = 150$ próbek. Dla każdego przedziału wyznaczono rozkłady prawdopodobieństwa dla rozkładu: normalnego, logarytmiczno-normalnego, Pareto i GEV metodą największej wiarygodności. Na rys. 8 przedstawiono wyniki oceny dopasowania uzyskanych rozkładów teoretycznych do rozkładów empirycznych czasów odpowiedzi serwerów przeprowadzone testem Kołmogorowa-Smirnowa, dla każdej z serii próbek. Wysokość słupka jest wartością p (4).

Cechą wspólną obydwu wykresów jest niejednoznaczność dopasowań i zmienność ich stopnia dopasowania mierzoną wartością p. Wynika to z dynamiki i złożoności procesów występujących w badanym zjawisku, co potwierdzają badania [15]. Ponadto na obydwu wykresach widać brak dopasowania do rozkładu Pareto. W przypadku serwera A2 pojawia się dopasowanie do rozkładu normalnego. Można przypuszczać, że szybsze procesy synchronizowanego dostępu do pamięci wbudowanej bazy danych będą się charakteryzować rozkładem normalnym. Widoczna różnica między wykresami to skala osi rzędnych. Widać, że dla serwera A2 są uzyskiwane lepsze dopasowanie rozkładów niż dla serwera A1. Lepsze dopasowania rozkładów dla serwera A2 wynikają prawdopodobnie z jego prostszej budowy, czyli braku zewnętrznej bazy danych, a tym samym braku procesów sieciowej komunikacji z nią. Rozkład czasu odpowiedzi serwera będzie zatem złożeniem mniejszej liczby rozkładów procesów działających współbieżnie w czasie pracy serwera.

Jeśli chodzi o parametry dopasowywanych rozkładów, to zostały zestawione w dwóch kolejnych tabelach. W tabeli 1 można zauważyć, że dla rozkładu GEV w Serii 2, gdzie mamy wysoki poziom istotności dopasowania, wartość średnia μ wynosi 2,17, co odpowiada wartości modalnej z histogramu dla serwera A1. Wartości średnie dla rozkładu log-normalnego mieszczą się w zakresie 2,30–3,02 ms i trudno dla nich znaleźć wartość modalną na histogramie.

Z tab. 2 wynika, że nie został zidentyfikowany rozkład dla wartości modalnej 0,75. W związku z tym przeprowadzono próbę dopasowania rozkładu dla zawężonego zakresu wartości czasów odpowiedzi 0,5–1,1 ze wszystkich próbek. Dla tego przypadku test zgodności wypadł pozytywnie jedynie dla rozkładu GEV uzyskując wartość $p = 0{,}0954$ z parametrami: $\xi = -0{,}0850$, $\mu = 0{,}7723$, $\sigma = 0{,}0927$. Uzyskana wartość średnia μ w dobrym przybliżeniu odpowiada identyfikowanej wartości modalnej. Obecność tak krótkich czasów wynika ze stosowanego w *WebSocket* mechanizmu wzajemnego odpytywanie między klientem a serwerem, polegającym na wysyłaniu bardzo krótkich komunikatów, na które druga strona musi jak najszybciej odpowiedzieć [19]. O obecności tych komunikatów może świadczyć wartość dominanty rozkładu rozmiaru pakietów z tab. 1. Inną przyczyną może być fragmentacja komunikatów, ale ze względu na bardzo małe rozmiary przesyłanych pakietów, zjawisko to nie powinno występować.



Rys. 8. Istotność statystyczna dopasowania rozkładów prawdopodobieństwa dla czasów opóźnień dla serwera A1 (z lewej) i serwera A2 (z prawej)
Fig. 8. Statistical significance of matching probability distributions for delay times for server A1 (left) and server A2 (right)

Tab. 2. Parametry rozkładów dla czasów odpowiedzi serwera A1
Tab. 2. Distribution parameters for A1 server response times

Seria	Log-normalny		GEV		
	μ	σ	ξ	μ	σ
1	2,3010	1,9438	0,2604	1,8527	1,1426
2	2,6574	1,8498	0,2203	2,1736	1,2374
3	3,0265	1,9152	0,0673	2,6214	1,5965
4	2,9772	2,0190	0,2420	2,4188	1,5711
5	2,4293	2,1289	0,2979	1,9322	1,3591
6	2,8080	1,9243	0,0449	2,4535	1,5041

Tab. 3. Parametry rozkładów dla czasów odpowiedzi serwera A2
Tab. 3. Distribution parameters for A2 server response times

Seria	Normalny		Log-normalny		GEV		
	μ	σ	μ	σ	ξ	μ	σ
1	0,4147	0,0923	0,4043	1,2557	–0,1731	0,3781	0,0862
2	0,4006	0,0808	0,3931	1,2117	–0,0248	0,3650	0,0646
3	0,4328	0,0822	0,4252	1,2063	–0,1022	0,3978	0,0725
4	0,4289	0,0704	0,4232	1,1771	–0,1394	0,3997	0,0635
5	0,4226	0,0702	0,4170	1,1764	–0,0756	0,3920	0,0598
6	0,4399	0,0961	0,4310	1,2166	0,0539	0,3973	0,0677

W przypadku parametrów statystycznych dla serwera A2 wszystkie rozkłady mają wartości średnie μ, które mieszczą się w bardzo bliskim zakresie od 0,37–0,43 i przedziałowo odpowiadają jego wartościom modalnym dla przedziału 0,32–0,46 (rys. 6). Niestety ich jednoznaczne przyporządkowanie jest niemożliwe. Współczynnik kształtu ξ dla rozkładu GEV jest praktycznie zbliżony i dla obydwu serwerów oscyluje wokół zera. Można zatem przyjąć przybliżenie, że dla obydwu serwerów jest on typu I, czyli jest rozkładem Gumbela.

4. Wnioski

W artykule przedstawiono propozycję optymalizacji serwera chmurowego dla sieciowych stanowisk symulatorów mieszanej rzeczywistości. Badania serwera przeprowadzono w laboratoryjnych warunkach sieci lokalnej. Na początkowym etapie prac serwer wykorzystywał bazę danych Redis oraz mechanizmy komunikacji *WebSocket* z protokołem *JSON*. W trakcie badań z wykorzystaniem architektury A1, zaobserwowano utratę

pakietów i niesatysfakcjonującą wydajność. W celu usunięcia tych wad zastosowano uproszczoną architekturę A2, w której wykorzystano protokół komunikacji z opartą na tablicach serializacją *MsgPack*. Optymalizację architektury osiągnięto poprzez wbudowanie bazy danych do serwera. Po wykonaniu pomiarów czasów odpowiedzi uzyskano wyniki potwierdzające zmniejszenie czasu odpowiedzi dla serwera A2, w stosunku do czasu odpowiedzi dla serwera A1 o rząd wielkości. Do oceny statystycznej czasów odpowiedzi, przeprowadzono identyfikację rozkładów prawdopodobieństwa i estymację ich parametrów wraz z oszacowaniem poziomu dopasowania. Do tego celu wykorzystano metodę największej wiarygodności oraz testy zgodności Kołmogorowa-Smirowa.

Dla obydwu serwerów uzyskano dopasowanie rozkładów czasów ich odpowiedzi do rozkładu Log-normalnego i Uogólnionego Wartości Maksymalnych (*GEV*). Dodatkowo dla serwera A2 uzyskano dopasowanie dla rozkładu normalnego. Podjęto próbę przybliżonej identyfikacji wartości modalnych histogramów. Niestety ich jednoznaczna identyfikacja w wielu przypadkach okazała się niemożliwa.

Bardzo duży wpływ na osiągane czasy odpowiedzi dla Serwera A1 miał miało niewątpliwie zastąpienie tekstowej serializacji *JSON* przez opartą na tablicach serializację *MsgPack*, o czym świadczą parametry statystyczne rozmiarów pakietów przedstawione w tab. 1.

Należy mieć na uwadze, że zaprezentowane podejście jest podejściem uproszczonym. Czas odpowiedzi systemów webowych zależy od obciążenia systemu oraz jego parametrów. Obciążenie może mieć wiele składowych reprezentujących miary obciążenia poszczególnych elementów systemu, zasobów [15] (np. bazy danych Redis w architekturze A1).

W celu dokładniejszej analizy statystycznej należałoby zidentyfikować elementy systemu i wykonać pomiary, w których pracowałyby jedynie wybrane elementy systemu, tak aby wyznaczyć statystyczny charakter ich wpływu na czas odpowiedzi. Interesującym wydaje się też zastosowanie w estymacji czasów odpowiedzi, modeli predykcji rozmytych szeregów czasowych [10].

Bibliografia

1. Bendat J.S., Piersol A.G., *Random Data – Analysis and Measurement Procedures*, Fourth Edition, John Wiley & Sons, Inc., Hoboken, New Jersey 2010.
2. Benjamini Y., De Veaux R.D., Efron B., Evans S., Glickman M., Graubard B.I., Kafadar K., *ASA President's Task Force Statement on Statistical Significance and Replicability*, "CHANCE", Vol. 34, No. 4, 2021, 10–11, DOI: 10.1080/09332480.2021.2003631.
3. Chojnacki D., *Parametryczna estymacja widma lokalnie stacjonarnych procesów losowych*, Rozprawa doktorska, Politechnika Gdańska, Gdańsk 2019.
4. Chakravarti I.M., Laha R.G., Roy J., *Handbook of Methods of Applied Statistics*, Vol. I, John Wiley and Sons, 1967.
5. Deliggnette-Muller M.L., Dutang Ch., *fitdistrplus: An R Package for Fitting Distributions*, "Journal of Statistical Software", Vol. 64, No. 4, 2015, DOI: 10.18637/jss.v064.i04.
6. Fisz W., *Rachunek prawdopodobieństwa i statystyka matematyczna*, wyd. IV, PWN, Warszawa 1969.
7. Golański P., Szczekala M., Roguszewski M., *Analysis of the 3D Object Reconstruction Accuracy in an Mi-17 Mixed Reality Simulator*, Automation 2021: Recent Achievements in Automation, Robotics and Measurement Technique, Vol. 1390, 2021, 136–145, DOI: 10.1007/978-3-030-74893-7_14.
8. Koszela J., Szymczyk M., *Rozproszona symulacja wirtualna – Chmura Symulacyjna*, „Symulacja w Badaniach i Rozwoju", Vol. 9, No. 1-2, 2018, 33–46.
9. Lu Y., Nolte T., Bate I., Cucu-Grosjean L., *A statistical Response-Time Analysis of Real-Time Embedded Systems*, 2012 IEEE 33rd Real-Time Systems Symposium, DOI: 10.1109/RTSS.2012.85.
10. Lucas P., Orang O., Silva P.C.L., Mendes E.M.A.M., Guimarães F., *A Tutorial on Fuzzy Time Series Forecasting Models: Recent Advances and Challenges*, "Learning and Nonlinear Models", Vol. 19, No. 2, 2022, 29—50, DOI: 10.21528/lnlm-vol19-no2-art3.
11. Łasocha W.P., Badurowicz M., *Porównanie wydajności protokołu WebSocket i HTTP*, „Journal of Computer Sciences Institute", Vol. 19, 2021, 67—74.
12. Ramakrishnan R., Kaur A., *Performance evaluation of web service response time probability distribution models for business process cycle time simulation*, "Journal of Systems and Software", Vol. 161, 2020, DOI: 10.1016/j.jss.2019.110480.
13. Son S.H., Lee I., Leung J.Y.-T., *Handbook of Real-Time and Embedded Systems*, Chapman and Hall/CRC, 2007.
14. Spinner S., Casale G., Brosig F., Kounev S., *Evaluating approaches to resource demand estimation*, "Performance Evaluation", Vol. 92, 2015, 51—71, DOI: 10.1016/j.peva.2015.07.005.
15. Zatwarnicki K., *Systemy webowe z jakością usług uwzględniające kryterium czasowe*, Problemy Projektowania, Politechnika Opolska ISSN 1429-6063, Opole 2012.

Inne źródła

16. https://www.json.org/json-en.html
17. https://msgpack.org/
18. https://hub.docker.com/
19. https://developer.mozilla.org/en-US/docs/Web/API/WebSockets_API/Writing_WebSocket_servers#pings_and_pongs_the_heartbeat_of_websockets
20. https://neuecc.medium.com/messagepack-for-c-v3-release-with-source-generator-support-893ed30d0e89

Statistical Assesment of Mixed-Reality Simulator Server Performance

Abstract: Despite the continuous increase in the computing power of modern computers, their capabilities are still insufficient in many areas. An example is the simulation of physical processes taking place in all types of training systems based on Virtual or Mixed-Reality technology. This is due to the growing requirements placed on these systems to achieve greater accuracy in reproducing reality. The solution to the above problem is the use of distributed simulation, which – by utilizing cloud solutions – allows for remote access to the computing power of specialized servers, and thus also provides the option of conducting remote training. In such a case, ensuring effective communication between the simulator's client stations and the server in real time is of utmost importance. This article presents a proposal for a server-client communication protocol, for use in the cloud version of the Mi-17 helicopter, Mixed-Reality simulator. A method for evaluating the performance of this protocol was proposed, along with a statistical description of the obtained results.

Keywords: cloud server, distributed simulation, mixed reality, fitting distributions, goodness of fit

dr inż. Piotr Golański
piotr.golanski@itwl.pl
ORCID: 0000-0002-3286-8678

Adiunkt w Zakładzie Integracji Systemów C4ISR Instytutu Technicznego Wojsk Lotniczych. Zajmuje się zagadnieniami związanymi z modelowaniem i symulacją procesów w systemach szkoleniowych.

Michał Golański
michal.golanski@itwl.pl
ORCID: 0009-0009-4836-3197

Student na Wydziale Mechatroniki, Uzbrojenia i Lotnictwa Wojskowej Akademii Technicznej. W Instytucie Technicznym Wojsk Lotniczych zajmuje się administracją systemów Linux, serwerami WWW oraz tworzeniem symulacji 3D.