

Analiza performansi sustava za udaljeno izvršavanje programskog kôda

Herman Zvonimir Došilović

Sveučilište u Zagrebu Fakultet elektrotehnike i računarstva

Zagreb, srpanj 2021.



Sadržaj

- 1 Arhitektura ekosustava sustava za udaljeno ocjenjivanje
 - Sustav za udaljeno izvršavanje programskog kôda
- 2 Analiza performansi sustava za udaljeno izvršavanje programskog kôda
- 3 Aplikacija Hélory
- 4 Primjer korištenja aplikacije Hélory
- 5 Zaključak



Arhitektura ekosustava sustava za udaljeno ocjenjivanje



Slika 1: Arhitektura OJ ekosustava. [2]

Arhitektura ekosustava sustava za udaljeno ocjenjivanje

Zaštićeno okruženje

```
#include </dev/random>

int main() {
    return 0;
}
```

Isječak 1: Ispravan C program s beskonačnim vremenom prevođenja.

```
#include <unistd.h>

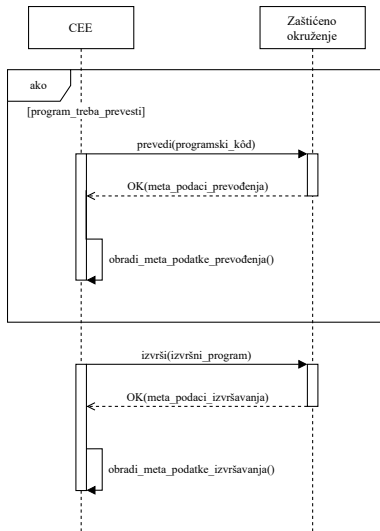
int main() {
    while(1) {
        fork();
    }
    return 0;
}
```

Isječak 2: C program s beskonačnim grananjem novih procesa.



Arhitektura ekosustava sustava za udaljeno ocjenjivanje

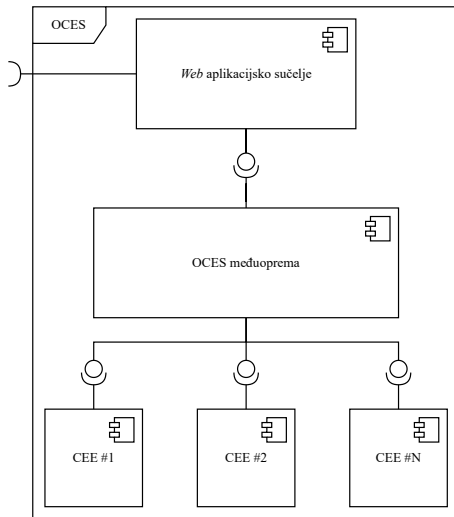
Modul za izvršavanje programskog kôda



Slika 2: Interakcija modula za izvršavanje programskog kôda s zaštićenim okruženjem.

Arhitektura ekosustava sustava za udaljeno ocjenjivanje

Sustav za udaljeno izvršavanje programskog kôda



Slika 3: Dijagram komponenti sustava za udaljeno izvršavanje programskog kôda.

Sustav za udaljeno izvršavanje programskog kôda

Sustav Sphere Engine



Slika 4: Vizualni identitet sustava Sphere Engine. [4]

Sustav za udaljeno izvršavanje programskog kôda

Sustav Piston



Slika 5: Vizualni identitet sustava Piston. [3]

Sustav za udaljeno izvršavanje programskog kôda

Sustav Judge0 (1)



Slika 6: Vizualni identitet sustava Judge0.

Sustav za udaljeno izvršavanje programskog kôda

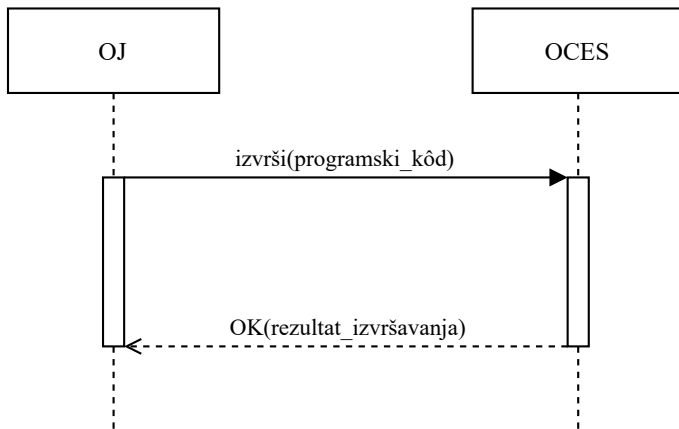
Sustav Judge0 (2)



Slika 7: Vizualni identiteti organizacija koje koriste sustav Judge0. [1]

Arhitektura ekosustava sustava za udaljeno ocjenjivanje

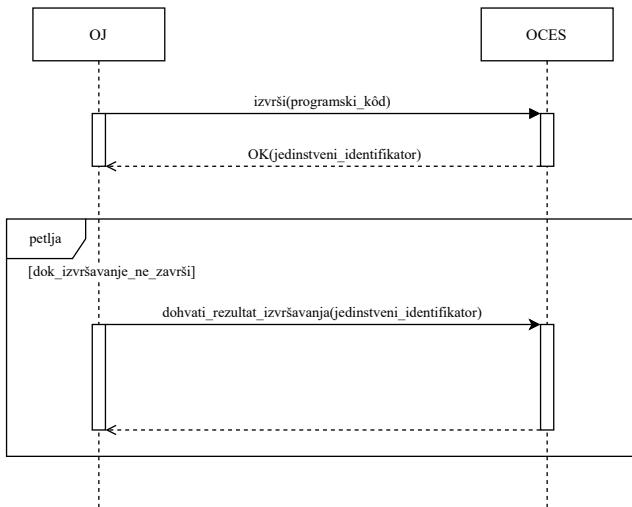
Sustav za udaljeno ocjenjivanje (1)



Slika 8: Sinkrona interakcija OJ-a i OCES-a.

Arhitektura ekosustava sustava za udaljeno ocjenjivanje

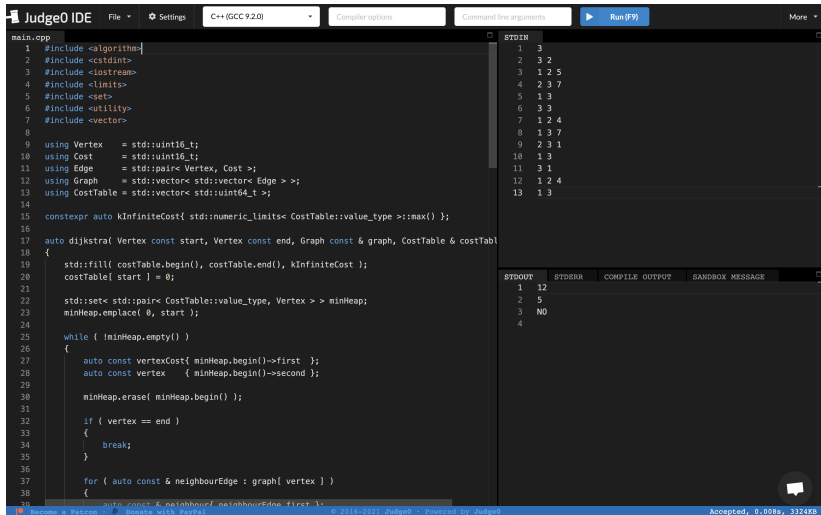
Sustav za udaljeno ocjenjivanje (2)



Slika 9: Asinkrona interakcija OJ-a i OCES-a.

Arhitektura ekosustava sustava za udaljeno ocjenjivanje

Web aplikacija za udaljeno programiranje



The screenshot displays the Judge0 IDE web application. The main editor shows a C++ file named `main.cpp` with the following code:

```
1 #include <algorithm>
2 #include <cstdint>
3 #include <iostream>
4 #include <limits>
5 #include <set>
6 #include <utility>
7 #include <vector>
8
9 using Vertex = std::uint16_t;
10 using Cost = std::uint16_t;
11 using Edge = std::pair<Vertex, Cost>;
12 using Graph = std::vector< std::vector< Edge > >;
13 using CostTable = std::vector< std::uint64_t >;
14
15 constexpr auto kInfiniteCost{ std::numeric_limits< CostTable::value_type >::max() };
16
17 auto dijkstra( Vertex const start, Vertex const end, Graph const & graph, CostTable & costTable )
18 {
19     std::fill( costTable.begin(), costTable.end(), kInfiniteCost );
20     costTable[ start ] = 0;
21
22     std::set< std::pair< CostTable::value_type, Vertex > > minHeap;
23     minHeap.emplace( 0, start );
24
25     while ( !minHeap.empty() )
26     {
27         auto const vertexCost{ minHeap.begin()->first };
28         auto const vertex { minHeap.begin()->second };
29
30         minHeap.erase( minHeap.begin() );
31
32         if ( vertex == end )
33         {
34             break;
35         }
36
37         for ( auto const & neighbourEdge : graph[ vertex ] )
38         {
39             auto const & neighbour{ neighbourEdge.first };
40             auto const & neighbourCost{ neighbourEdge.second };
41             auto const & currentCost{ costTable[ vertex ] + neighbourCost };
42             if ( currentCost < costTable[ neighbour ] )
43             {
44                 costTable[ neighbour ] = currentCost;
45                 minHeap.emplace( currentCost, neighbour );
46             }
47         }
48     }
49 }
```

The terminal window on the right shows the following output:

```
STDIN
1 3
2 3 2
3 1 2 5
4 2 3 7
5 1 3
6 3 3
7 1 2 4
8 1 3 7
9 2 3 1
10 1 3
11 3 1
12 1 2 4
13 1 3

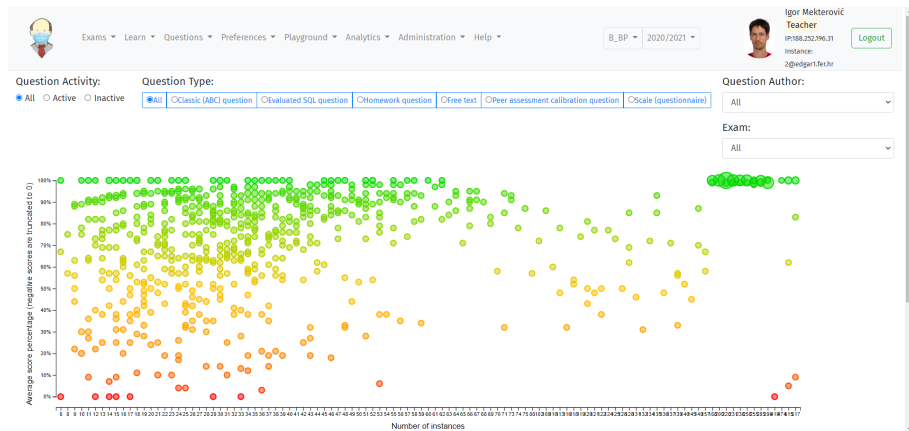
STDOUT
1 12
2 5
3 NO
4
```

The status bar at the bottom indicates: "Accepted, 0.008s, 3324KB".

Slika 10: Sučelje web aplikacije Judge0 IDE.

Arhitektura ekosustava sustava za udaljeno ocjenjivanje

Specijalizirane izvedbe sustava za udaljeno ocjenjivanje (1)



Slika 11: Sučelje web aplikacije Edgar iz perspektive učitelja.

Arhitektura ekosustava sustava za udaljeno ocjenjivanje

Specijalizirane izvedbe sustava za udaljeno ocjenjivanje (2)

The screenshot displays the CodeChum web application interface. At the top, the CodeChum logo is visible with a 'Premium' badge. A navigation bar includes a 'Create' button, a moon icon, a bell icon, a user profile for 'Maranga', and a help icon. Below this, a banner encourages users to 'Help us spread fun and interactive programming education to your fellow teachers and earn P1000 per invite.' The main content area is titled 'My Classes > PSITE Python Class'. The class details include a Python logo, the title 'PSITE Python Class' with an 'Active' status, and metadata: 'Python', 'SAT 4:00PM - 7:00PM', '138 students', and 'Class Code: climax-559'. A tabbed interface shows 'Announcements', 'Timeline' (selected), 'Activities', 'Scores', 'Students', and 'Files'. The 'Timeline' tab displays two lesson cards: 'How To Print' (Output - 5 Topics, 30% of students have finished) and 'Variables' (Storing Data Into Variables - 4 Topics, 21% of students have finished). A sidebar on the left contains links for 'Activities', 'Classes', 'Problems', and 'Playground'. A contact box at the bottom left provides information for 'Jude, CEO'.

CODECHUM Premium

+ Create

Help us spread fun and interactive programming education to your fellow teachers and earn P1000 per invite. Send Invite

My Classes > PSITE Python Class

PSITE Python Class Active

Python SAT 4:00PM - 7:00PM 138 students Class Code: climax-559

Announcements **Timeline** Activities Scores Students Files

How To Print
Output - 5 Topics
30% of students have finished
View Lesson

Variables
Storing Data Into Variables - 4 Topics
21% of students have finished

Have any questions or concerns? Contact me at (+63) 995 946 9610 - Jude, CEO

Slika 12: Sučelje web aplikacije CodeChum.

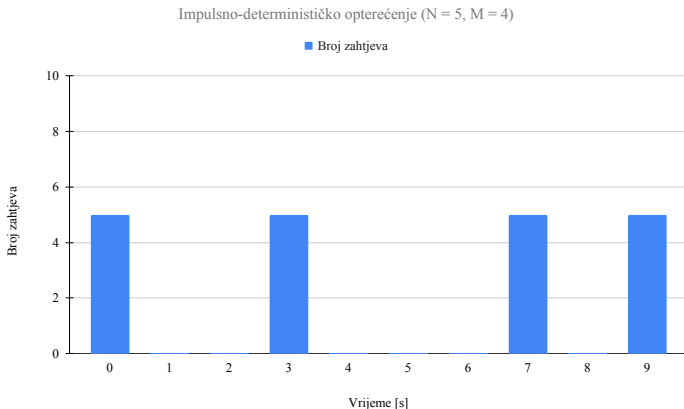
Analiza performansi sustava za udaljeno izvršavanje programskog kôda

- dinamike višekorisničkog opterećenja
 - ▶ impulsno-determinističko opterećenje
 - ▶ kontinuirano-determinističko opterećenje
 - ▶ kontinuirano-stohastičko opterećenje
- strategije izvršavanja
 - ▶ jednostavan scenarij
 - ▶ scenarij procesorskog opterećenja
 - ▶ scenarij mrežnog opterećenja
 - ▶ scenarij procesorskog i mrežnog opterećenja
- metrike za analizu performansi
 - ▶ uspješnost izvršavanja
 - ▶ vrijeme obrade
 - ▶ maksimalno opterećenje



Analiza performansi sustava za udaljeno izvršavanje programskog kôda

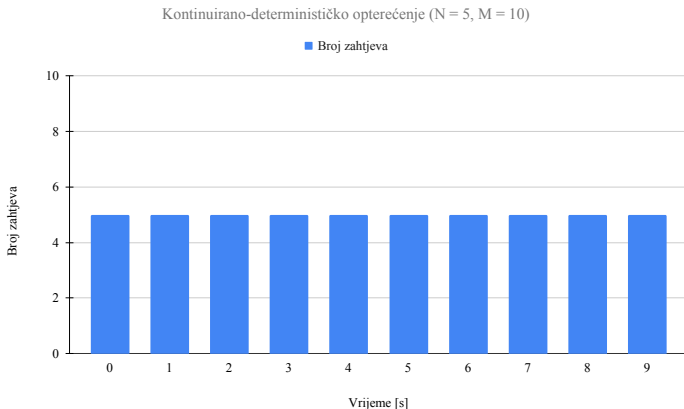
Impulsno-determinističko opterećenje sustava



Slika 13: Impulsno-determinističko opterećenje sustava ($N = 5$, $M = 4$).

Analiza performansi sustava za udaljeno izvršavanje programskog kôda

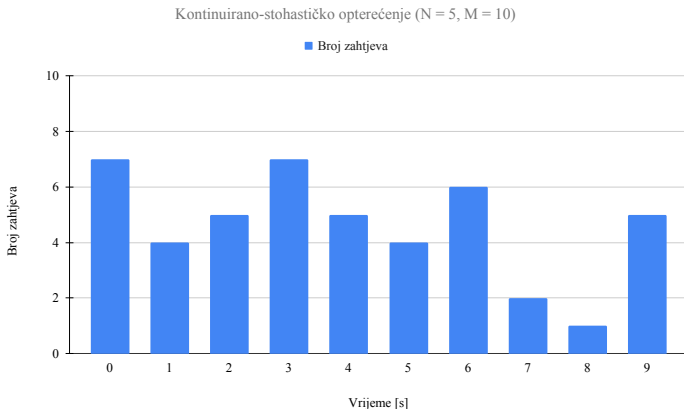
Kontinuirano-determinističko opterećenje sustava



Slika 14: Kontinuirano-determinističko opterećenje sustava ($N = 5$, $M = 10$).

Analiza performansi sustava za udaljeno izvršavanje programskog kôda

Kontinuirano-stohastičko opterećenje sustava



Slika 15: Kontinuirano-stohastičko opterećenje sustava ($N = 5$, $M = 10$).

Analiza performansi sustava za udaljeno izvršavanje programskog kôda

Scenariji korištenja i programski jezici

Scenariji korištenja:

- jednostavan scenarij
- scenarij procesorskog opterećenja
- scenarij mrežnog opterećenja
- scenarij procesorskog i mrežnog opterećenja

Programski jezici:

- C++
- Java
- Python



Analiza performansi sustava za udaljeno izvršavanje programskog kôda

Metrike za analizu performansi

- uspješnost izvršavanja
- vrijeme obrade
- maksimalno opterećenje



Aplikacija Hélyory

Pregled komandno-linijskog sučelja

```
$ ./helory run deterministic --users 5x10 \  
                                --scenario hello_world \  
                                --endpoint judge0_ce
```

Isječak 3: Primjer pokretanja impulsno-determinističkog opterećenja.

```
$ ./helory run deterministic --users 5x10 --no-wait \  
                                --scenario cpu_intensive \  
                                --endpoint judge0_ce_fer
```

Isječak 4: Primjer pokretanja kontinuirano-determinističkog opterećenja.

```
$ ./helory run stochastic --intensity 5 --duration 10 \  
                                --scenario cpu_and_network_intensive \  
                                --endpoint piston_public
```

Isječak 5: Primjer pokretanja kontinuirano-stohastičkog opterećenja.

Aplikacija H lory

Pregled su elja grafi kog izvještaja (1)

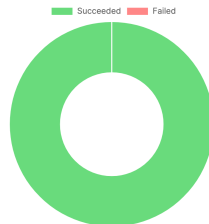


H lory Benchmark Report

[Download as PDF](#)

Scenario	Hello World		
Description	Plain "hello, world" program.		
Language	C++		
Endpoint	Judge0 CE		
Strategy	async		
Type	stochastic		
Configuration	duration	10	
	intensity	5	
	seed	1624807317443342000	
Started At	27/06/2021, 17:21:57		
Finished At	27/06/2021, 17:22:11		
Duration	00:00:14		

Success Rate



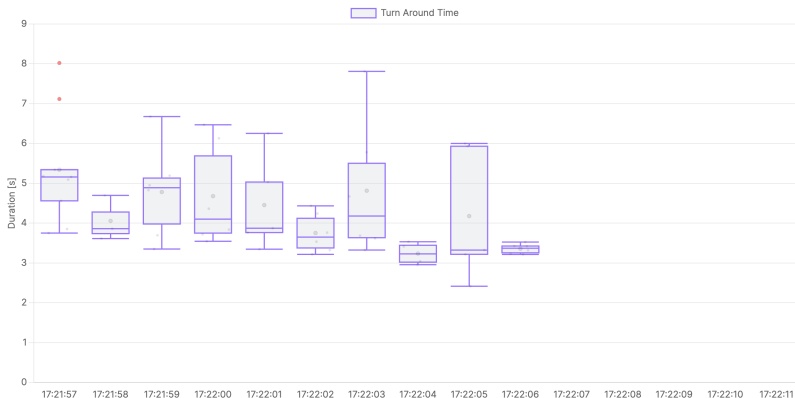
Interval 1s

Slika 16: Prikaz detalja o pokrenutom eksperimentu.

Aplikacija Hélyory

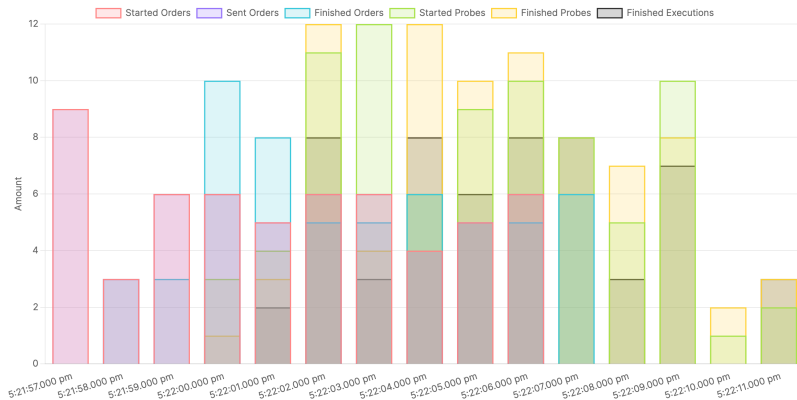
Pregled sučelja grafičkog izvještaja (2)

Turn Around Times

[Download Chart Data](#)

Slika 17: Prikaz dijagrama vremena obrade zahtjeva.

Order and Probe Requests

[Download Chart Data](#)

Slika 18: Prikaz grafa zahtjeva narudžbe i zahtjeva ispitivanja.

Aplikacija Hélyory

Pregled sučelja grafičkog izvještaja (4)

Raw Benchmark Data

[Download](#)

```
{
  "id": "2021-06-27T17:21:57+02:00-stochastic-hello_world-cpp-judge0_ce-async-int_5-dur_10",
  "name": "Hello World in C++ via Judge0 CE with stochastic behavior.",
  "started_at": "2021-06-27T17:21:57.443642+02:00",
  "finished_at": "2021-06-27T17:22:11.449519+02:00",
  "scenario": "Hello World",
  "scenario_description": "Plain \"hello, world\" program.",
  "language": "C++",
  "endpoint": "Judge0 CE",
  "endpoint_url": "https://ce.judge0.com",
  "strategy": "async",
  "experiment_type": "stochastic",
  "experiment_configuration": {
    "duration": 10,
    "intensity": 5,
    "seed": 1624807317443342000
  },
  "executions": [
    {
      "success": true,
      "started_at": "2021-06-27T17:21:57.444496+02:00",
      "finished_at": "2021-06-27T17:22:01.194987+02:00",
      "order_started_at": "2021-06-27T17:21:57.444718+02:00",

```

Generated by [Hélyory](#) on Sun Jun 27 2021

Slika 19: Prikaz sirovih podatak prikupljenih za vrijeme eksperimenta.

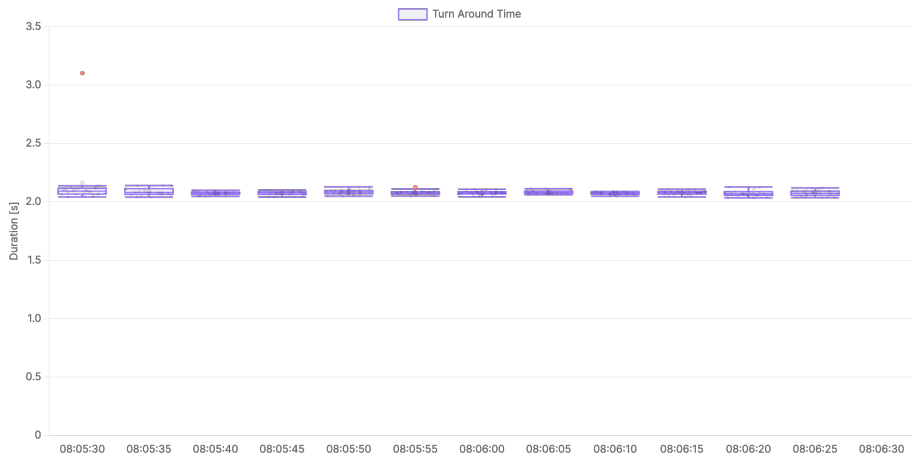
Primjer korištenja aplikacije Hélyory

- > 500 eksperimenata
- FER-ova instanca sustava Judge0
- kontinuirano-stohastičko opterećenje
 - ▶ $N \in \{1, 5, 10, 15, 20, 25\}$
 - ▶ $M = 60$
- jednostavan scenarij korištenja
- sinkrona i asinkrona interakcija
- C++, Java i Python



Primjer korištenja aplikacije Hélyory

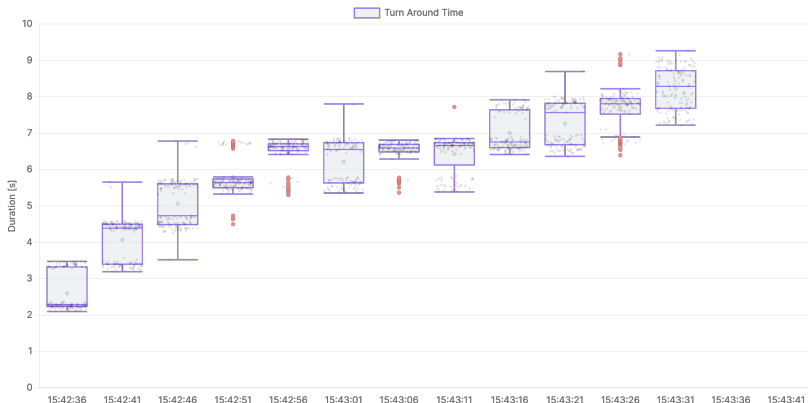
Analiza performansi FER-ove instance sustava Judge0 (1)



Slika 20: Vrijeme obrade zahtjeva FER-ove instance sustava Judge0 za $N = 5$ i $M = 60$ s asinkronom interakcijom.

Primjer korištenja aplikacije Hélorý

Analiza performansi FER-ove instance sustava Judge0 (2)

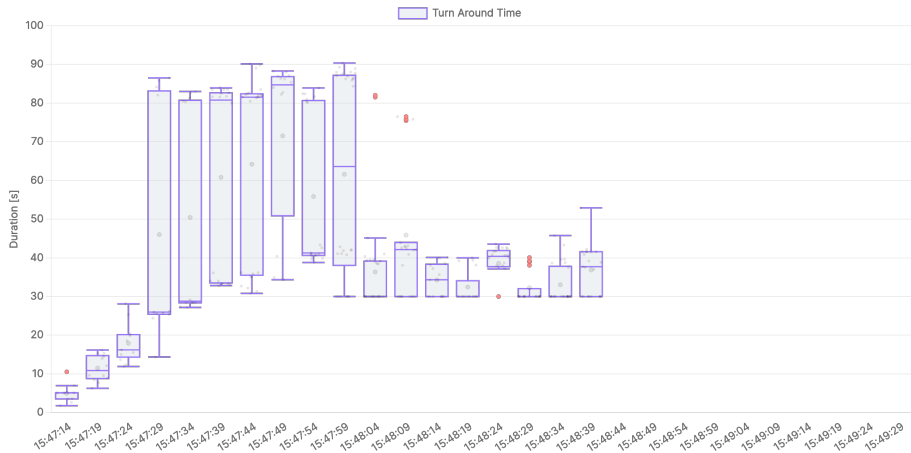


Slika 21: Vrijeme obrade zahtjeva FER-ove instance sustava Judge0 za $N = 25$ i $M = 60$ s asinkronom interakcijom.

Primjer korištenja aplikacije Hélorý

Analiza performansi FER-ove instance sustava Judge0 (3)

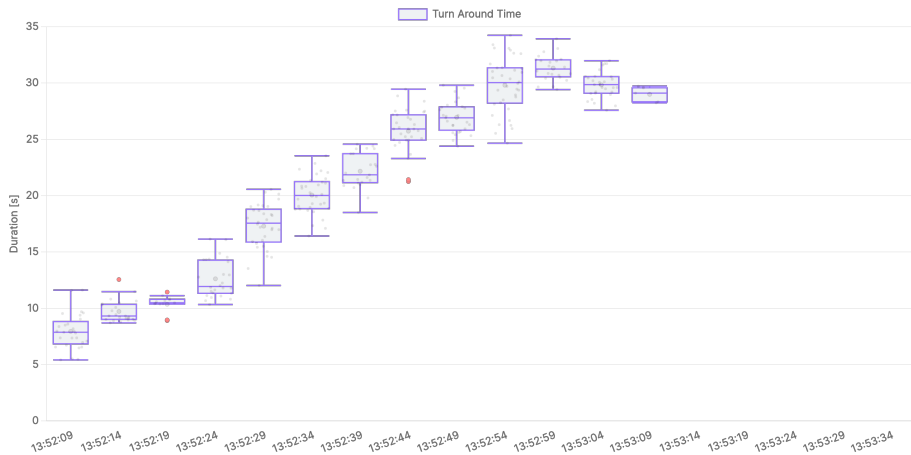
- uspješnost obrade: 70,21%



Slika 22: Vrijeme obrade zahtjeva FER-ove instance sustava Judge0 za $N = 5$ i $M = 60$ sa sinkronom interakcijom.

Primjer korištenja aplikacije Hélogy

Analiza performansi FER-ove instance sustava Judge0 (4)



Slika 23: Vrijeme obrade zahtjeva FER-ove instance sustava Judge0 za $N = 5$ i $M = 60$ scenarija procesorskog opterećenja u Java implementaciji.

Primjer korištenja aplikacije H  lory

Analiza performansi FER-ove instance sustava Judge0 (5)

Programski jezik	Scenarij korištenja	
	hello_world	cpu_intensive
C++	20	1
Java	10	1
Python	25	1

Tablica 1: Maksimalno opterećenje koje podnosi FER-ova instanca sustava Judge0.



Zaključak

- OCES-i imaju ključni utjecaj na korisničko iskustvo.
- Prvi **radni okvir** za analizu performansi i ocjenu kvalitete i pouzdanosti usluge koju nude OCES-i.
- Aplikacija **Hélory** koja implementira predstavljeni radni okvir.
- Eksperimentalno je dokazano da se pri korištenju sustava Judge0 preporuča koristiti asinkronu interakciju sa sustavom.
- Eksperimenti nad FER-ovom instancom sustava Judge0 pokazuju dobre rezultate.



- [1] Herman Zvonimir Došilović. Judge0 - Where code happens., 2020. URL <https://judge0.com>. Pristupano: 25.06.2021.
- [2] Herman Zvonimir Došilović i Igor Mekterović. Robust and Scalable Online Code Execution System. U *2020 43rd International Convention on Information, Communication and Electronic Technology (MIPRO)*, stranice 1627–1632, 2020. doi: 10.23919/MIPRO48935.2020.9245310.
- [3] Brian Seymour. A high performance general purpose code execution engine., 2018. URL <https://github.com/engineer-man/piston>. Pristupano: 20.06.2021.
- [4] Sphere Research Labs Sp. z o.o. Coding skills assessment and code execution APIs - Sphere Engine, 2008. URL <https://sphere-engine.com>. Pristupano: 25.06.2021.