|  |  |  |
| --- | --- | --- |
|  | Univerzitet u Novom Sadu  **FAKULTET TEHNIČKIH NAUKA**  **NOVI SAD** |  |

**Napredni industrijski komunikacioni protokoli u elektroenergetskim sistemima**

Projektni zadatak 9

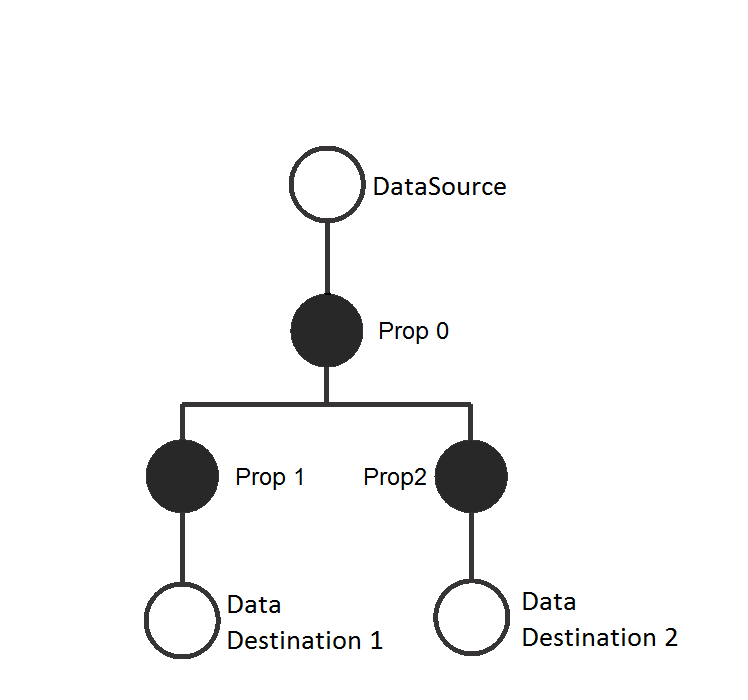
Studenti:

Igor Spremo E5 ?/2015

Nenad Dragišić E5 1/2015

Novi Sad, 2015.

# 1. DIZAJN DOKUMENTA



Slika 1 – Struktura sistema za propagaciju podataka

Sistem se sastoji od:

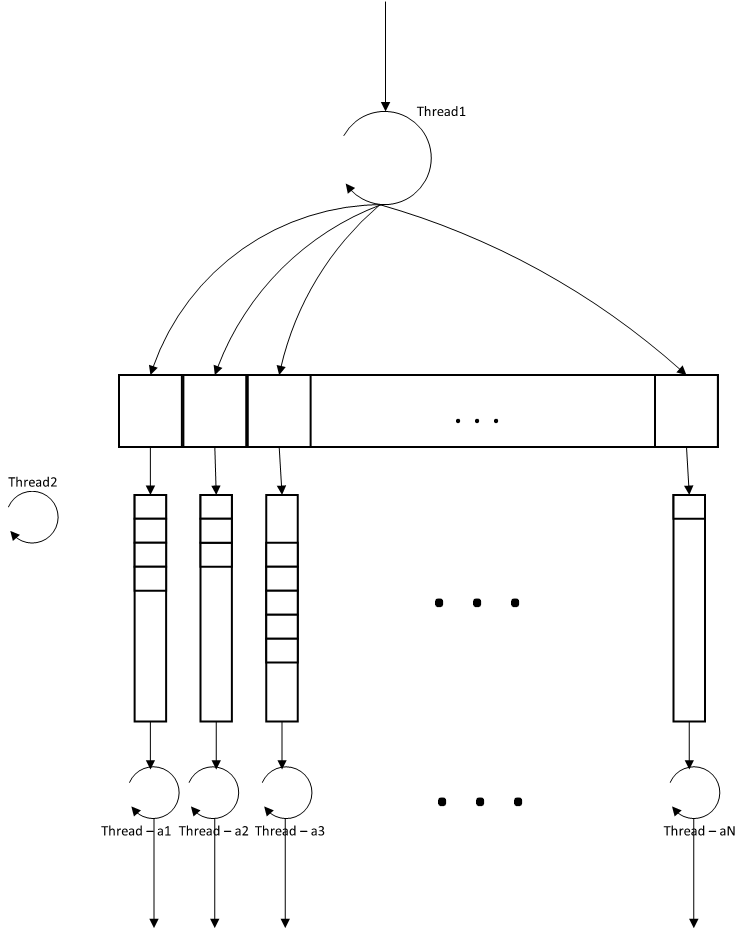
1. DataSource-a – Služi za generisanje podataka i prosleđivanja istih ka svojim podelementima
2. Propagator-a – Njegova jedina svrha je da prosledi primljene podatke ka svojim podelementima (može ih biti više)
3. DataDestination-a – Predstavlja krajnji servis kojem su podaci koje DataSource šalje od interesa (može ih biti više)

Propagator se sastoji od niti i niza struktura koja sadrži: kružni bafer za skladištenje primljenih podataka (thread safe), niti koja konzumira podatke i propagira ih ka odgovarajućem podelementu i SOCKET koji omogućava pronalaženje podelementa za slanje. Za svaki podelement koji se poveže za odgovarajući nadelement se u nadelementu kreira nova instanca strukture, koja se inicijalizuje i dodaje u niz.

Na slici 2 se može videti dizajn propagatora. Thread1 predstavlja nit koja čeka da se neki drugi propagator ili DataDestination poveže na njega. Kada se to desi, funkcija ove niti je realizovana tako da za novi priključeni čvor kreira instancu ThreadSafeQueue-a i doda mu consumer nit (zatim se ova struktura smešta u niz). Thread2 je nit koja po prijemu podataka sa roditeljske strane smešta isti podatak u svaki kružni bafer svih podelemenata. Thread-a1-aN su consumer niti. Njihova svrha je vađenje podataka iz kružnog bafera i prosleđivanje odgovarajućem podčvoru za koji je zadužena prilikom priključenja istog.

Aplikacija je testirana na sledecim performansama:

1. Procesor: Intel Pentium CPU J2900 @ 2.41 GHz
2. Ram memorija: 8 GB
3. Operativni sistem: 64-bit Windows 10



Slika 2 – Dizajn propagatora

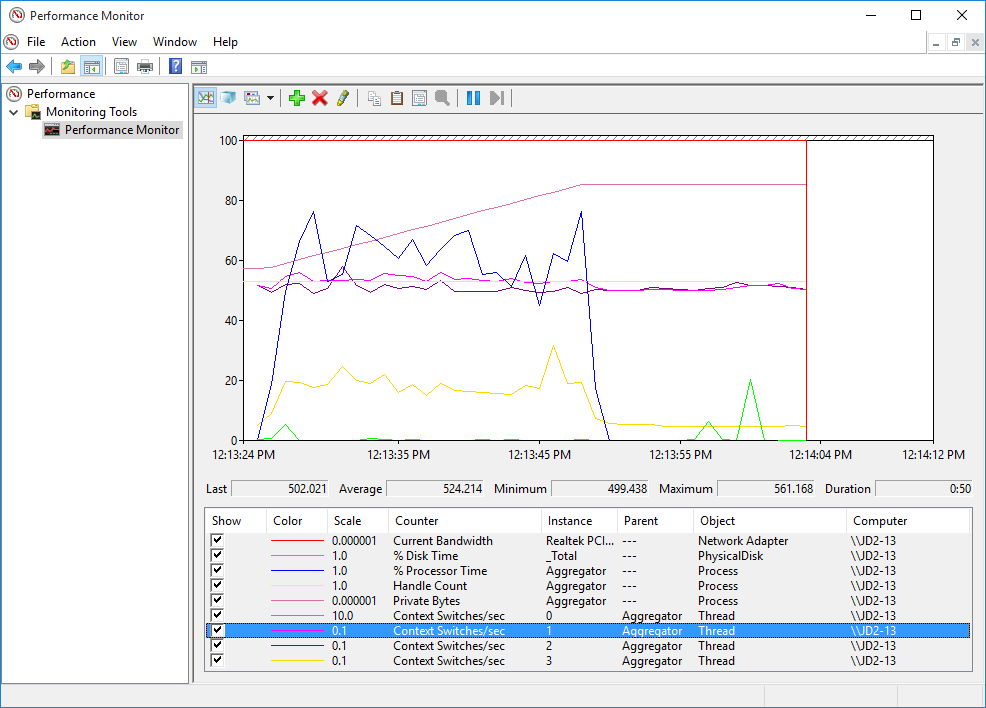
# 2. Korišćene strukture podataka

Korišćene strukture podataka su:

1. Kružni bafer – Pogodna struktura za producer – consume šablon jer se elementi smeštaju u bafer samo prilikom dodavanja i prilikom promene veličine istog. Bolja struktura od liste jer sadrži informaciju o prvom podatku koji treba da se čita, kao i informacija gde treba da se smesti sledeći, bez potrebe iteriranja. U ovom slučaju u bafer je smeštana struktura koja sadrži dužinu podataka i pokazivač na podatke od interesa.
2. Niz sa ograničenom dužinom – Služi za čuvanje pokazivača ka strukturi koja sadrži kružni bafer, consumer nit i socket potreban za prosleđivanje podataka. Krajnji cilj je bio da se napravi dinamički niz, ali se od te ideje odustalo jer je naiđeno na probleme prevezivanja pokazivača, kao parametar, funkcijama za niti. Ovim smo ograničili broj podelemenata koji mogu da se povežu na određeni nadelement.

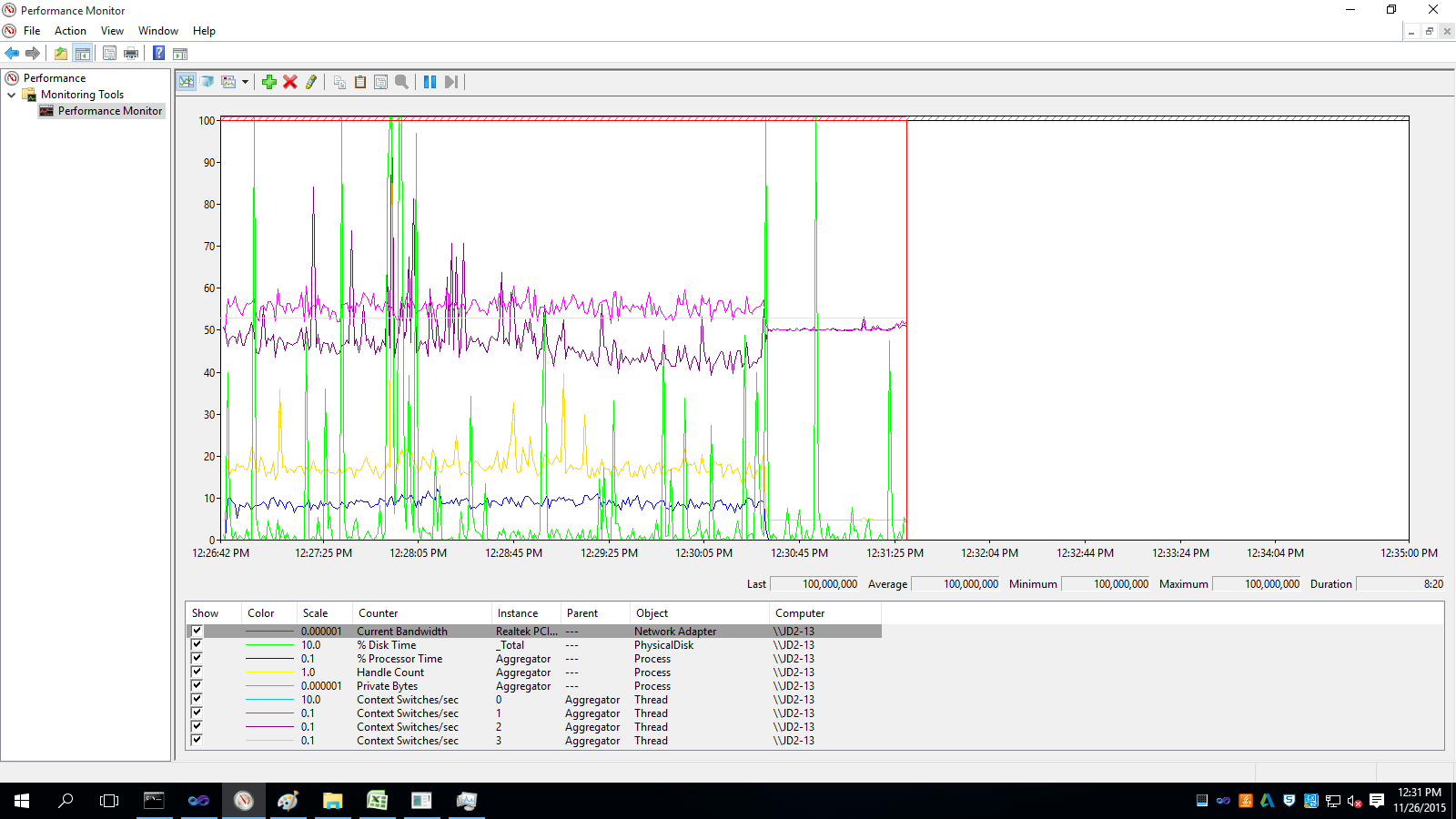
# 3. Rezultati testiranja

Prilikom testiranja uspešno su poslati podaci od 100 MB, 1 GB i 10 GB. Poruke koje su slane su bile dužine 1 KB.



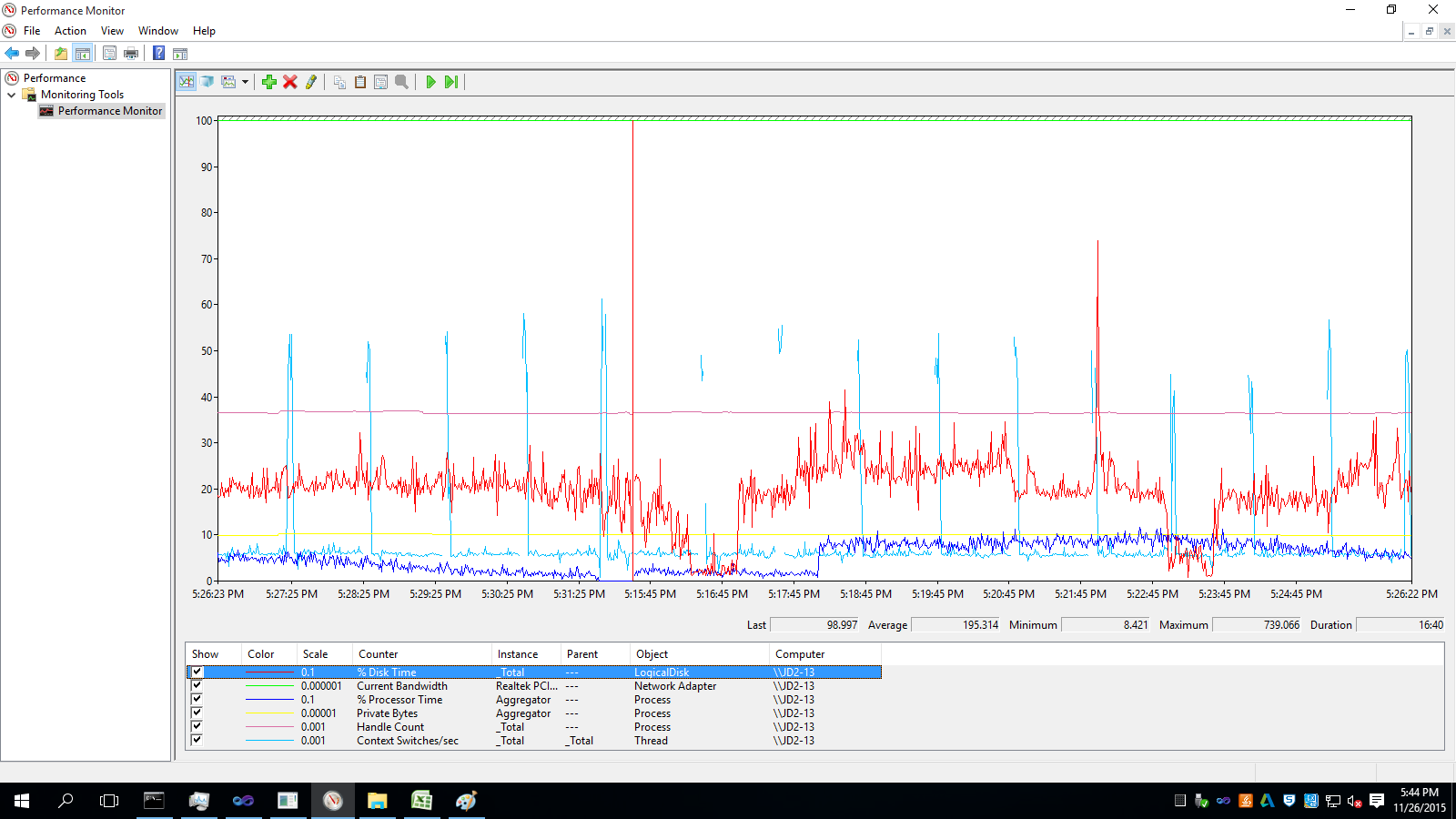
Slika 3 – Izgled grafikona prilikom slanja podataka od 100 MB

Na slici 3 se vidi da je iskorišćenje procesorskog vremena tokom slanja podataka od 100 MB oko 70%. Takođe je iskorišćenje mreže maksimalno. Context Switch-evi su između 200 i 530 što je normalno. Broj Handle-ova je oko 55 i konstantno je do gašenja aplikacije. Vreme potrebno za prenos ove količine podataka je između 20 i 25 sekundi.



Slika 4 - Izgled grafikona prilikom slanja podataka od 1 GB

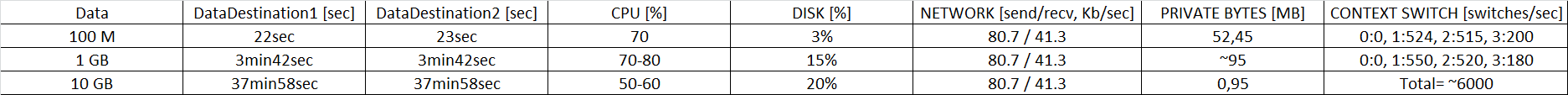
Na slici 4 se vidi da je iskorišćenje procesora između 70 i 80%. Context Switch-evi se kreći između 180 i 550. Broj Handle-ova je oko 55 i konstantno je do gašenja aplikacije. Vreme potrebno za prenos ove količine podataka je oko 3 min i 40 sekundi.



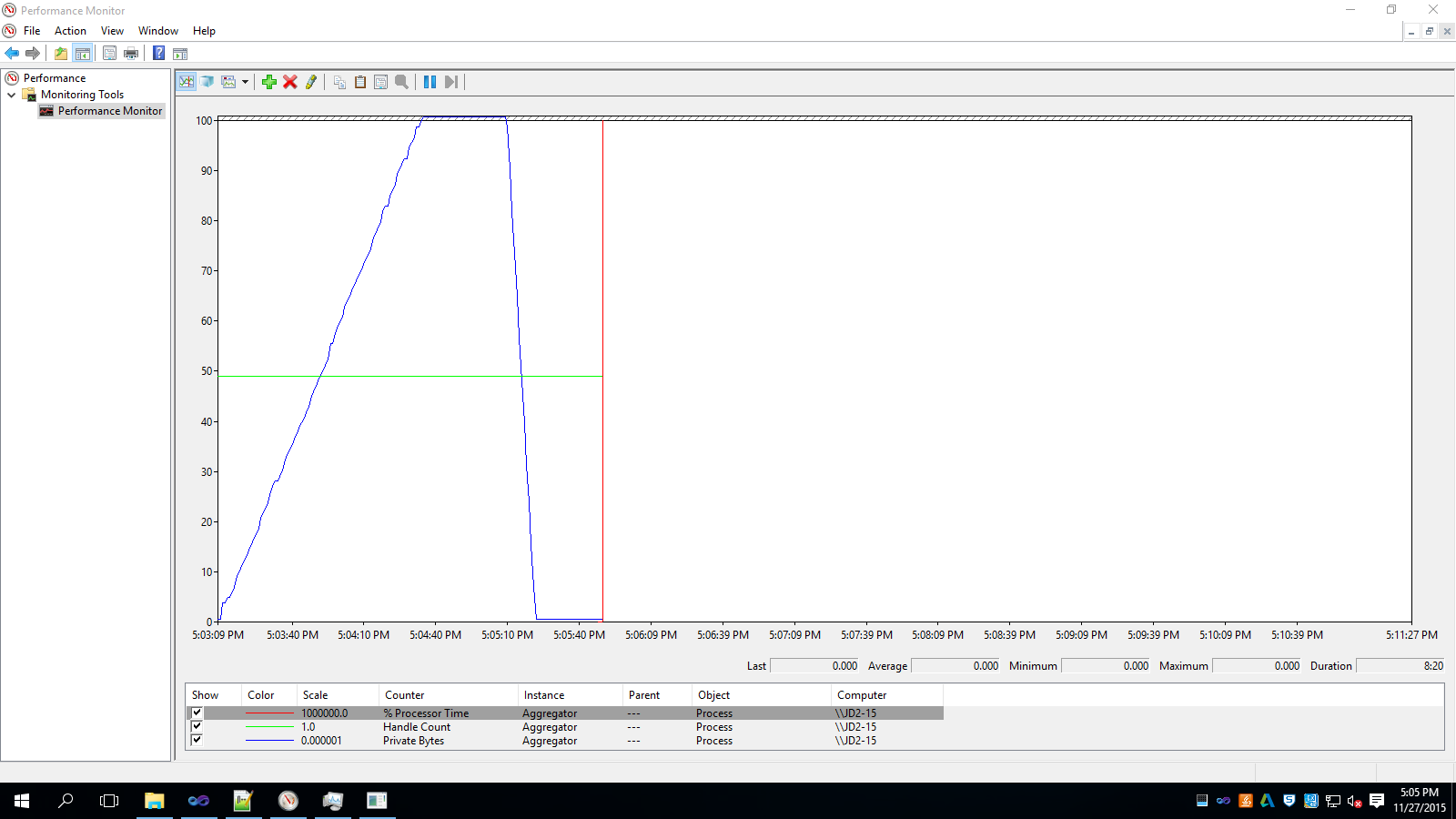
Slika 5 - Izgled grafikona prilikom slanja podataka od 10 GB

Na slici 5 se vidi grafikon koji pokazuje iskorišćenje performansi tokom slanja 10 GB podataka. Opterećenje procesora je između 50 i 60%. Vreme potrebno za prenos ove količine podataka je oko 37 minuta i 58 sekundi.

Na slici 6 se vide vrednosti merenja u proseku.



Slika 6 – Tabela sa prosečnim vrednostima merenja



Slika 7 – prikaz zauzeća i oslobađanja memorije

Na slici 7 se može videti kako prilikom početka slanja podataka zauzeće memorije raste, a prilikom završetka slanja se memorija oslobađa. Zauzeće memorije raste zbog povećanja kapaciteta kružnog bafera, a oslobađanje se vrši prilikom njegovog smanjivanja.

# 4. Potencijalna unapređenja rešenja

Jedno od mogućih unapređenja je da se niz ograničene dužine zameni dinamičkim nizom. To bi nam donelo mogućnost priključivanja neograničenog broja podčvorova nadčvoru. U trenutnoj situaciji mora se programski odrediti maksimalan broj podelemenata i on je nepromenjiv tokom izvršavanja programa.

Umesto čuvanja pokazivača na podatke u kružnom baferu, mogu se čuvati podaci što bi ubrzalo rad jer bi bilo manje alokacija i dealokacija memorije.

Prilikom promene veličine da se ne kopira stari sadržaj for petljom nego sa memcpy. To bi dovelo do dodatnog ubrznanja izvršavanja.