**VISOKO UČILIŠTE ALGEBRA**
ZAGREB, REPUBLIKA HRVATSKA

**Dominik Despot**

**NUMA arhitektura**

Seminarski rad

Zagreb, svibanj 2023.

# Sadržaj

[2. Uvod 1](#_Toc136799641)

[3. Princip rada 2](#_Toc136799642)

[4. Prednosti NUMA arhitekture 3](#_Toc136799643)

[4.1. Brzina 3](#_Toc136799644)

[4.2. Skalabilnost 3](#_Toc136799645)

[4.3. Smanjena konkurencija za memoriju 3](#_Toc136799646)

[4.4. Fleksibilnost u smještaju podataka 3](#_Toc136799647)

[5. Nedostatci 4](#_Toc136799648)

[5.1. Povećana latencija pristupa udaljenoj memoriji 4](#_Toc136799649)

[5.2. Neuravnotežena opterećenja 4](#_Toc136799650)

[5.3. Kompleksnost 4](#_Toc136799651)

[5.4. Ograničenje kapaciteta 4](#_Toc136799652)

[6. Scenarij: Prekomjerna alokacija memorije u virtualnoj mašini 5](#_Toc136799653)

[6.1. Posljedice prekomjerne alokacije memorije virtualnoj mašini 5](#_Toc136799654)

[6.1.1. Povećani pristup udaljenoj memoriji 6](#_Toc136799655)

[6.1.2. Neravnoteža opterećenja 6](#_Toc136799656)

[6.1.3. Interferencija s drugim virtualnim strojevima 6](#_Toc136799657)

[6.1.4. Memory paging/swapping 6](#_Toc136799658)

[6.1.5. Rušenje sustava 6](#_Toc136799659)

[6.2. Rješenje 7](#_Toc136799660)

[6.2.1. Prevencija 7](#_Toc136799661)

[6.2.2. Prilagođivanje dodijeljene memorije 7](#_Toc136799662)

[6.2.3. Nadograđivanje sustava 7](#_Toc136799663)

[7. Zaključak 8](#_Toc136799664)

# Sažetak

Ovaj rad pruža analizu arhitekture Non-Uniform Memory Access (NUMA) i njezine implikacije za sustave s više procesora. Istražuje prednosti i nedostatke NUMA arhitekture. Dodatno, rad ispituje scenarij u kojem je virtualnoj mašini dodijeljena veća memorija nego što procesor na kojem se izvršava ima spojen na svoj memorijski kontroler, raspravljajući o posljedicama i predlažući moguća rješenja za ublažavanje problema.

# Uvod

Rastuća potreba za računalnim sustavima visokih i širokih performansi dovela je do razvoja višeprocesorskih arhitektura. NUMA, ili Non-Uniform Memory Access, je arhitektura računalne memorije korištena kod takvih, višeprocesorskih sustava, pri kojom vrijeme pristupa memoriji ovisi o lokaciji memorije relativno od procesora. U NUMA sustavima procesor može brži pristupiti svojoj lokalnoj memoriji nego ne-lokalnoj memoriji (poput lokalnoj memoriji drugog procesora ili zajedničkoj memoriji). Prednosti NUMA arhitekture ponajviše ovise o primjeni. Najizraženije su kod serverskih sustava gdje su velike količine podataka presudne za rad.

# Princip rada

NUMA arhitektura se sastoji od više čvorova procesora, od kojih svaki ima vlastitu lokalnu memoriju i kontroler memorije. Ti čvorovi su povezani putem brze veze te tako omogućujući optimiziranu komunikaciju i balansiran pristup memoriji između procesora.

# Prednosti NUMA arhitekture

## Brzina

Podjelom memorije u NUMA sustavima, u radnim zadacima gdje se izvođenje oslanja na lokalne podatke procesori će brže pristupati njima jer će se nalaziti u njihovoj lokalnoj memoriji i tako će se smanjiti vrijeme pristupa memoriji te će se stoga programi brže izvoditi.

## Skalabilnost

NUMA arhitektura omogućuje jednostavnije skaliranje višeprocesorskih sustava. Ovaj pristup izbjegava uske grlo (engl. bottleneck) povezan s centraliziranim sustavom memorije, omogućujući bolje performanse kako broj procesora raste.

## Smanjena konkurencija za memoriju

Pružajući lokalnu memoriju svakom procesoru, NUMA smanjuje konkurenciju nad memorijom jer procesori mogu pristupati svojoj lokalnoj memoriji s manjom latencijom.

## Fleksibilnost u smještaju podataka

NUMA arhitektura omogućuje kontrolu pozicioniranja podataka u memoriji, optimizirajući performanse osiguravanjem da često korišteni podaci budu smješteni u lokalnu memoriju.

# Nedostatci

## Povećana latencija pristupa udaljenoj memoriji

Pristup udaljenoj memoriji u arhitekturi NUMA ima veću latenciju zbog dodatnih troškova komunikacije, što može utjecati na ukupne performanse sustava.

## Neuravnotežena opterećenja

Nejednolika raspodjela poslova na procesore može rezultirati neravnotežom opterećenja, što dovodi do suboptimalnog korištenja resursa.

## Kompleksnost

NUMA sustavi zahtijevaju pažljivo upravljanje memorijom i strategije smještaja podataka kako bi se postigle optimalne performanse, što ih čini složenijima za dizajniranje i upravljanje u usporedbi s arhitekturama uniformnog pristupa memoriji.

## Ograničenje kapaciteta

Ukoliko su potrebe za memorijom veće od dodijeljenih, proizaći će problemi u konfliktu s memorijom drugog/drugih procesora te će se program usporiti.

# Scenarij: Prekomjerna alokacija memorije u virtualnoj mašini

## Posljedice prekomjerne alokacije memorije virtualnoj mašini

U scenariju u kojem je virtualnom stroju dodijeljena više memorije nego što procesor ima spojen na svoj memorijski kontroler, mogu se pojaviti sljedeće posljedice:

* Povećani pristup udaljenoj memoriji
* Neuravnotežena opterećenja
* Interferencija s drugim virtualnim strojevima
* *Memory paging/swapping*
* Rušenje sustava

U svakom slučaju ovakvi scenariji se nastoje izbjegavati radi neefikasnosti i problema koje stvaraju.

### Povećani pristup udaljenoj memoriji

Višak memorije virtualnog stroja može biti smješten u udaljenoj memoriji, što dovodi do povećanja latencija i smanjenja performansi zbog čestih pristupa udaljenoj memoriji.

### Neuravnotežena opterećenja

Neravnoteža između kapaciteta procesora i dodijeljene memorije može uzrokovati neravnotežu opterećenja, što utječe na ukupne performanse sustava i iskorištenje resursa.

### Interferencija s drugim virtualnim strojevima

Prekomjerna alokacija memorije može dovesti do sukoba i interferencije s drugim virtualnim strojevima koji dijele isti NUMA sustav, utječući i na njihove performanse.

### Memory paging/swapping

Ukoliko nema više raspoložive radne memorije, operacijski sustav će biti prisiljen dodijeliti komade trajne memorije u svrhu radne memorije, što će dodatno usporiti performanse.

### Rušenje sustava

Zadnja očekivana posljedica je rušenje sustava. „*Out of memory*“ greške u principu i jesu najčešći uzrok rušenja sustava te se stoga isto može očekivati i u ovom scenariju kao posljednja posljedica.

## Rješenje

Za rješavanje problema mogu se implementirati razna rješenja.

### Prevencija

U svakom slučaju, najbitniji korak je prevencija takvih scenarija, kako se navedene greške ne bi pojavljivale.

### Prilagođivanje dodijeljene memorije

Ukoliko primijetimo da se pojavljuju problemi s našom virtualnom mašinom potrebno je prilagoditi dodijeljenu memoriju kako bismo zaobišli prethodno navedene probleme.

### Nadograđivanje sustava

Ukoliko su nam zahtjevi stvarno iznad raspoloživih resursa, jedina preostala opcija jest nadograđivanje sustava ili smanjenje zadataka na procesoru.

# Zaključak

NUMA arhitektura pruža snažan okvir za postizanje visokih performansi u sustavima s više procesora. Iako donosi značajne prednosti, pažljivo razmatranje potencijalnih izazova i implementacija odgovarajućih rješenja, poput dinamičkog balansiranja memorije ključni su kako bi se u potpunosti iskoristile prednosti NUMA arhitekture i osigurala optimalna izvedba sustava.

**Popis literature**

[1] Lim, Geunsik; Suh, Sang-Bum; User-Level Memory Scheduler for Optimizing Application Performance in NUMA-Based Multicore Systems

[2] Dakić, Vedran; INFLUENCE OF VIRTUAL STORAGE CONTROLLER TYPE ON MICROSOFT SQL SERVER 2019 PERFORMANCE

[3] Wikipedia, „[Non-uniform memory access](https://en.wikipedia.org/wiki/Non-uniform_memory_access)“, pristupljeno 4. lipnja 2023.