УНИВЕРЗИТЕТ У БЕОГРАДУ
МАТЕМАТИЧКИ ФАКУЛТЕТ

Наталија Драшковић

ПРОНАЛАЖЕЊЕ ПЕРИОДА НАЈМАЊЕ АКТИВНОСТИ ПРОЦЕСОРА У ОКРУЖЕЊУ ОБЛАКА ПРИМЕНОМ МЕТОДА АНАЛИЗЕ ВРЕМЕНСКИХ СЕРИЈА

мастер рад

Београд, 2023.

Ментор:

др Александар Картељ, доцент
Универзитет у Београду, Математички факултет

Чланови комисије:

др Владимир Филиповић, редовни професор
Универзитет у Београду, Математички факултет

др Бојана Милошевић, ванредни професор
Универзитет у Београду, Математички факултет

Датум одбране:

Наслов мастер рада: Проналажење периода најмање активности процесора у окружењу облака применом метода анализе временских серија

Резиме: Често је у окружењу облака потребно извршити операције које могу негативно утицати на искуство корисника јер изазивају привремену недоступност или смањење перформанси ресурса које корисник користи, на пример повремено ажурирање и одржавање система. Овакве акције ометају рад корисника и могу неповољно утицати на задовољство корисника облак окружењем, због чега се стално размишља о могућностима за унапређивање избора тренутка када ће се дате операције извршавати, као и могућностима за њихово што боље планирање. Идеје за решавање оваквог проблема базиране су на идеји да пружаоци услуга у облаку покушају да на што бољи начин предвиде начин на који корисници користе ресурсе, а затим операције које могу утицати на корисника извршавају у периодима када је корисничка активност минимална. Пружаоцима услуга у облаку најчешће нису познати детаљи начина на који корисници користе ресурсе, те је један од првих корака дефиниција релевантне величине која може добро описати меру употребе ресурса од стране корисника у датом тренутку. Испоставља се да постоји тесна веза између мере активности процесора виртуелних машина које корисник користи и активности корисника у том тренутку, што омогућава да се проблем проналажења тренутка у коме ће одржавање најмање утицати на корисника сведе на проблем проналажења периода најмање активности процесора. Активност процесора током времена може се посматрати као једна временска серија, односно овај проблем се даље може свести на проблем анализе временских серија. Посматрајући рад процесора као једну временску серију приказани резултати ће бити засновани на анализи временских серија. Описивање временске серије биће засновано на карактеристикама као што су аутокорелација, стационарност, тренд и сезоналност, а за предвиђање њених будућих вредности користићемо ARIMA моделе.

Кључне речи: рачунарство у облаку, окружење облака, периоди најмање активности, анализа временских серија, ARIMA модел

Садржај

Глава 1

Увод

Фокус овог рада биће дефинисање ефикасних начина за проналажење периода најмање активности процесора у окружењу облака у сврху коришћења те информације приликом планирања будућих одржавања окружења облака. Примена метода анализе временских серија има значајну улогу [11] јер омогућава проучавање и моделовање активности процесора и препознавање најмање активних периода посматрајући активност процесора током времена као једну временску серију.

У области рачунарства у облаку, где се ресурси деле између више корисника, оптимално управљање ресурсима је од веома великог значаја [13]. Проналажење периода са најмањом активношћу процесора има потенцијал да допринесе ефикаснијем коришћењу ресурса и оптимизацији рада у облаку. Идеја која ће бити представљена у овом раду односи се на проналажење најбољег периода за одржавање облак окружења, где се најбољим периодом за одржавање окружења облака сматра онај период у коме ће утицај одржавања на корисника бити минималан. Међутим, слични резултати засновани на посматрању активности процесора могли би се употребити и за решавање проблема смањења потрошње енергије или повећања искоришћења хардверских ресурса, где се даље могу разматрати могућности за аутоматске промене величине инстанци облака или активацију, односно деактивацију резервних ресурса, као и других проблема из области управљања ресурсима у облак окружењу.

Описаћемо и користити методе анализе временских серија као што су аутокорелација, ауторегресија, као и креирање модела који би на најбољи начин описали дату временску серију, са фокусом на сезоналним ARIMA моделима.

Циљ ће нам бити проналажење одређених узорака понашања и периодичности активности процесора, што ће нам омогућити да предвиђања која будемо правили верно описују посматране временске серије.

У глави 2 дефинисан је појам временске серије и уведене су неопходне математичке дефиниције из ове области које ће бити потребне у наставку. Поред дефиниција описани су и циљеви анализе временских серија, типови временских серија, као и њихове битне компоненте. Након дефинисања свих неопходних појмова дефинисани су и математички модели који могу описати временске серије и који су коришћени у глави 4 приликом имплементације, као и поступак за изградњу ARIMA модела који су дефинисали Бокс[1] и Џенкинс[2].

У глави 3 уведен је појам рачунарства у облаку. Најпре је дефинисано шта је то рачунарство у облаку и његове стандардне карактеристике, а затим су описане врсте облака, модели пружања услуга у облаку, као и архитектура облака. Након дефинисања основних информација које су нам потребне за разумевање рачунарства у облаку, дефинисани су појмови битни за дефинисање проблема избора најбољег времена за одржавање окружења облака, као што су уговор о нивоу услуге и периоди одржавања.

Глава 4 фокусирана је на решавање проблема проналажења периода најмање активности процесора применом метода анализе временских серија и имплементацију тог решења у програмском језику *Python*. Описан је скуп података над којим је рађено, модели коришћени за његов опис и предвиђање будућих вредности, као и визуелне репрезентације појединих временских серија и добијених резултата.

На крају, у глави 5 дат је кратак закључак на основу разматрања и резултата добијених у претходним главама овог рада.

[1]George Box, Британски статистичар
[2]Gwilym Jenkins, Британски статистичар

Глава 2

Анализа временских серија

Временска серија представља низ података уређених према времену, обично посматраних у једнаким временским интервалима. Сваки податак у временској серији повезан је са одређеним временским тренутком и представља вредност посматране величине у датом тренутку. Временске серије срећу се у могим областима, укључујући економију, финансије, метеорологију, индустрију, телекомуникације и многе друге. Примери временских серија могу бити различите величине, као што су дневне температуре, месечни приходи неког предузећа, недељна продаја неког производа, степен искоришћености одређених ресурса итд. Како је процес доношења одлука често повезан са предвиђањем будућих вредности променљивих које зависе од времена, временске серије и њихова анализа представљају погодно средство за добијање ових информација.

Анализа временских серија је статистичка дисциплина која је пронашла примену у многим областима, због чега бележи динамичан развој. Ова дисциплина подразумева процес разумевања и описивања података који су прикупљени током времена и може укључивати различите методе и технике, као што су линеарна регресија, ARIMA модели, машинско учење (нпр. неуронске мреже), спектрална анализа и друге технике за моделовање, предвиђање и интерполацију података из временских серија. Циљ анализе временских серија је идентификација образаца понашања, трендова, сезоналности и других битних карактеристика посматраних података како би се донели информисани закључци и одлуке.

2.1 Случајни процеси

Формалну дефиницију временске серије можемо дати са становишта случајног процеса. Како се до појма случајног процеса долази се проширивањем појма случајне променљиве биће неопходно да најпре уведемо следеће дефиниције [24, 25, 22].

Дефиниција 2.1.1 ***Скуп исхода*** *једног експеримента означава се са* Ω. *Елементи овог скупа означавају се* $\omega_i, i = 1, 2, 3, ...$ *и називају се* ***елементарни догађаји***.

Дефиниција 2.1.2 *Подскуп* $\mathcal{A}$ *партитивног скупа* $\mathcal{P}(\Omega)$ *је* ***алгебра*** *догађаја над* Ω *ако важе следећа својства:*

- $\Omega \in \mathcal{A}$;
- $A \in \mathcal{A} \Rightarrow \overline{A} \in \mathcal{A}$;
- $A \in \mathcal{A} \wedge B \in \mathcal{A} \Rightarrow A \cup B \in \mathcal{A}$.

Дефиниција 2.1.3 *Подскуп* $\mathcal{A}$ *партитивног скупа* $\mathcal{P}(\Omega)$ *је* σ***-алгебра*** *догађаја над* Ω *ако важе следећа својства:*

- $\Omega \in \mathcal{A}$;
- $A \in \mathcal{A} \Rightarrow \overline{A} \in \mathcal{A}$;
- $\{A_i\}_{i \in \mathbf{N}} \in \mathcal{A} \Rightarrow \bigcup_{i=1}^{n} A_i \in \mathcal{A}$.

Дефиниција 2.1.4 *Нека је* $\mathcal{K}$ *колекција подскупова исхода* Ω, *при чему сама колекција није* σ*-алгебра. Пресек свих* σ*-алгебри које садрже колекцију је једна* σ*-алгебра и она се назива* ***минимална*** σ***-алгебра*** *генерисана колекцијом и означава се са* $\sigma[\mathcal{K}]$.

Дефиниција 2.1.5 *Нека је скуп исхода* Ω *једнак скупу реалних бројева* $\mathbf{R}$ *и нека је колекција дата са* $\mathcal{K} = \{(a, b] : a, b \in \mathbf{R}, a < b\}$. *Минимална* σ*-алгебра која садржи колекцију зове се* ***Борелова*** σ***-алгебра*** *подскупова реалне праве. Означава се са* $\mathcal{B}$. *Њени елементи се називају Борелови скупови на реалној правој.*

Дефиниција 2.1.6 *Нека је Ω скуп исхода случајног експеримента и $\mathcal{A}$ σ-алгебра догађаја над Ω. Уређени пар $(\Omega, \mathcal{A})$ назива се* ***мерљив простор****.*

Дефиниција 2.1.7 *Нека је Ω скуп исхода свих елементарних догађаја и $\mathcal{A}$ σ-алгебра догађаја над Ω. Функција $P : \mathcal{A} \to \mathbf{R}$ назива се* ***вероватноћа*** *или вероватносна мера на мерљивом простору $(\Omega, \mathcal{A})$ ако има следећа својства:*

- *нормираност: $P(\Omega) = 1$;*
- *ненегативност: $P(A) \geq 0$, за сваки догађај $A \in \mathcal{A}$;*
- *σ-адитивност вероватноће: ако $\{A_i\}_{i \in \mathbf{N}} \in \mathcal{F}, A_i \cap A_j = \emptyset, i \neq j, i = 1, 2, 3, ..., j = 1, 2, 3, ...,$ онда $P\left(\bigcup_{i=1}^{\infty} A_i\right) = \sum_{i=1}^{\infty} P(A_i)$.*

Дефиниција 2.1.8 *Уређена тројка $(\Omega, \mathcal{A}, P)$ назива се* ***простор вероватноћа*** *или вероватносни модел посматраног експеримента.*

Дефиниција 2.1.9 *Нека је $(\Omega, \mathcal{A}, P)$ простор вероватноћа и нека је $(\mathcal{S}, \mathcal{B})$ мерљив простор. Тада се функција $X : \Omega \to \mathbf{R}$ која је мерљива у односу на σ-алгебре $\mathcal{A}$ и $\mathcal{B}$, у смислу да за сваки скуп $B \in \mathcal{B}$ важи $X^{-1}(B) = \{\omega : X(\omega) \in B\} \in \mathcal{A}$ назива* ***случајни елемент*** *у простору $\mathcal{S}$. Простор $(\mathcal{S}, \mathcal{B})$ назива се простор случајног елемента X.*

Дефиниција 2.1.10 *Ако у претходној дефиницији важи да је $S = \mathbf{R}$ и ако је $\mathcal{B}$ Борелова σ-алгебра подскупова скупа $\mathbf{R}$, онда је функција X* ***случајна променљива****.*

До појма случајног процеса долази се проширивањем појма случајног елемента. Случајни елемент је функција која као резултат даје могуће исходе случајног експеримента. Ако се случајни елемент посматра у зависности од времена, тада он постаје и функција времена. У општем случају, случајни елемент не зависи од времена, али многе појаве и догађаји, чији су исходи случајни, а који се одвијају у времену, захтевају да се појам случајног елемента уопшти тако да се укључи и временска компонента t. Посматрјући фамилију случајних елемената који зависе од времена, долази се до дефиниције стохастичког или случајног процеса.

Дефиниција 2.1.11 *Фамилија случајних елемената* $\{X_t(\omega), t \in T, \omega \in \Omega\}$ *дефинисаних над истим простором вероватноће* $(\Omega, \mathcal{A}, P)$*, где је* T *бесконачан скуп назива се стохастички или* ***случајни процес****. Скуп* T *назива се индексни или параметарски скуп.*

Скуп T из претходне дефиниције одређује тип случајног процеса. Уколико је скуп T неки интервал из скупа реалних бројева $\mathbf{R}$, тада се параметар $t \in T$ најчешће интерпретира као време, па је случајан процес $\{X_t, t \in T\}$ случајан процес са непрекидним временом. Ако је скуп T неки подскуп скупа целих бројева $\mathbf{Z}$, тада је случајан процес $\{X_t, t \in T\}$ случајан процес са дискретним временом или случајни низ.

За фиксирано $t \in T$, функција X_t је једна случајна променљива. Свака случајна променљива X_t има свој закон расподеле, који је одређен одговарајућом функцијом расподеле $F_{X_t}(x) = P\{X_t < x\} = P\{\omega \in \Omega : X_t(\omega) < x\}$.

Нека је фиксирано n временских тренутака $t_1, t_2, ..., t_n$. Сваком од тих тренутака одговра по једна случајна променљива. Тако добијамо n случајних променљивих $X_{t_1}, X_{t_2}, ..., X_{t_n}$ које се могу посматрати као координате неког коначнодимензионог случаног вектора $(X_{t_1}, X_{t_2}, ..., X_{t_n})$. У општем случају, неопходно је знати и расподеле случајних вектора.

Дефиниција 2.1.12 *За сваки природан број* n *и произвољне елементе* $t_1, t_2, ..., t_n \in T, t_1 < t_2 < ... < t_n$*, расподела вероватноћа случајног вектора* $(X_{t_1}, X_{t_2}, ..., X_{t_n})$ *одређена је функцијом расподеле* $F_{X_{t_1,...,t_n}}(x_1, ..., x_n) = P\{X_{t_1} \leq x, ..., X_{t_n} \leq x\}$*. Фамилија свих ових расподела назива се фамилија коначнодимензионих расподела случајног процеса* $\{X_t, t \in T\}$*.*

Фамилија коначнодимензионих расподела случајног процеса задовољава следеће услове:

- услов симетрије – ако је $i_1, i_2, ..., i_n$ једна пермутација бројева од 1 до n, тада важи $F_{X_{t_{i_1},...,t_{i_n}}}(x_1, ..., x_n) = F_{X_{t_1,...,t_n}}(x_1, ..., x_n)$;
- услов сагласности – ако је $m < n$ за произвољне $t_{m+1}, t_{m+2}, ..., t_n \in T$ важи $F_{X_{t_1,...,t_m,t_{m+1},...,t_n}}(x_1, ..., x_m, \infty, ..., \infty) = F_{X_{t_1,...,t_m}}(x_1, ..., x_m)$.

Нека је $\{X_t, t \in T\}$ случајан процес. Тада важи:

- средња вредност случајног процеса износи $E(X_t) = \mu_t, t \in T$;
- дисперзија случајног процеса износи $D(X_t) = \sigma_t^2, t \in T$;
- аутоковаријациона функција случајног процеса износи $\gamma_x(r,s) = cov(X_r, X_s) = E((X_r - E(X_r))(X_s - E(X_s))), r, s \in T$;
- аутокорелациона функција случајног процеса износи $\rho_X(r,s) = \frac{cov(X_r,X_s)}{\sqrt{D(X_r)D(X_s)}}$, $r, s \in T$.

Дефиниција 2.1.13 *Случајан процес* $\{X_t, t \in T\}$ *је* ***строго стационаран*** *ако су његове коначнодимензионе функције расподеле инваријантне у односу на транслацију времена* t*, односно ако за све* $t_i + t \in T, i = 1, 2, ...$ *важи* $F_{X_{t_1,...,t_n}}(x_1, ..., x_n) = F_{X_{t_1+t,...,t_n+t}}(x_1, ..., x_n)$.

Дефиниција 2.1.14 *Случајан процес* $\{X_t, t \in T\}$ *је* ***слабо стационаран*** *ако су испуњени следећи услови:*

- $E(X_t) = m = const, \forall t \in T$;
- $E(X_t^2) < \infty, \forall t \in T$;
- $cov(X_r, X_s) = cov(X_{r+t}, X_{s+t}) = E((X_{r+t} - m)(X_{s+t} - m)), \forall r, s, t \in T$.

Дефиниција 2.1.15 ***Временска серија*** *је случајан процес индексиран по времену.*

Однос временске серије и случајног процеса одговара односу узорка и основног скупа у теорији статистичког закључивања. Као што узорак представља део основног скупа на основу кога се доносе закључци о самом скупу, тако и временска серија представља део случајног процеса који треба да омогући уочавање карактеристика тог процеса.

Код анализе временских серија битне су статистичке особине временске серије као случајног процеса, па се као основне особине узимају средња вредност (очекивање) и дисперзија (варијанса). Такође, једно од основних оруђа за алализу временско зависних опсервација, какве су оне у временској серији је аутокорелациона функција. Њоме је исказана корелациона структура временске серије, односно узајамна зависност две опсервације временске серије које су међусобно удаљене за известан временски период.

2.2 Циљеви анализе временских серија

Праћење неке појаве током времена је основа проналажења логике у следу опсервација, откривању узрока проблема, као и у разумевању и контролисању саме појаве и њеног узрока. Када говоримо о анализи временске серије говоримо о могућности да се објасни и предвиди наредни корак у нумеричкој реализацији неке појаве тако да он не одступа сувише од праве вредности. У том смислу, анализа временских серија има четири парцијална циља који заједно чине један основни циљ који подразумева добијање жељеног одговора на основу познатих података [23]:

- **Описивање** – У циљу изучавања основних карактеристика временске серије користимо њене графичке приказе и сумарне статистике. Опис кључних карактеристика временске серије има вредност сам по себи, али најчешће представља прву етапу анализе временске серије. Наиме, често се током анализе покаже да је већ једноставним графичким приказом дошла до изражаја суштинска карактеристика посматране појаве, те да нема потребе позивати у помоћ сложеније статистичке методе анализе временских серија. Само описивање временске серије представља одређивање информација о природи временске серије и о томе да ли је потребно извршити неку трансформацију временске серије пре наставка саме анализе.

- **Објашњење** – У овој фази анализе временских серија циљ је изабрати одговарајући статистички модел који описује кретање временске серије на задовољавајући начин. Када располажемо са више серија могуће је користити варијације једне временске серије у циљу објашњења варијација у другој временској серији. Избор модела зависи од тога да ли је у питању једнодимензиона или вишедимензиона анализа. Генерализацијом модела једнодимензионих временских серија, као што су на пример ауторегресиони модели, можемо добити моделе вишедимензионих временских серија, тзв. векторске ауторегресионе моделе.

- **Прогнозирање** – На основу прошлих опсервација идентификује се и оцењује изабрани модел временске серије који се потом користи за формирање прогнозе будућих вредности временске серије. У свакој од етапа изградње модела користи се хетерогени скуп статистичких тестова и

критеријума којима се верификује ваљаност коришћеног модела у поређењу са другим конкурентним моделима.

- **Контрола** – Добре прогнозе омогућавају формирање модела функције преноса временске серије, у циљу контролисања самог посматраног процеса. Потребно је формирати задовољавајући модел функције у смислу да резултујући процес буде што ближи жељеном циљу.

2.3 Типови временских серија и њихове компоненте

Временске серије можемо класификовати коришћењем различитих критеријума. Једна подела је на **временске серије са непрекидним и дискретним временом**. Временска серија са непрекидним временом је она временска серија код које опсервације можемо регистровати у ма ком временском тренутку. Временска серија са дискретним временом је она серија код које опсервације бележимо у одређеним временским тренуцима, најчешће у истим временским интервалима (дневно, месечно, квартално или годишње).

Особина непрекидности временске серије проистиче из природе посматране појаве. До временске серије са дискретним временом можемо доћи и на основу временске серије са непрекидним временом, тако што ћемо регистровање посматране појаве вршити само у одређеним временским интервалима. Тада кажемо да смо систематским узорком, у једнаким временским интервалима, бележили вредности временске серије са непрекидним временом. Пример тако добијене временске серије може бити цена акција на берзи. Цена акција варира непрекидно током дана, али се њена вредност бележи само у тренутку затварања берзе. Други начин добијања временске серије са дискретним временом је временским агрегирањем посматране временске серије са непрекидним временом у изабраном интервалу времена. Пример овако добијених временских серија може бити вредност увоза или извоза неког производа у одређеном временском периоду.

Временске серије се још могу поделити и на **стационарне** и **нестационарне**. Једна од веома битних особина сваке временске серије јесте њена стационарност. Временска серија ће бити стационарна уколико је њено кретање предвидиво током времена, односно уколико њено кретање има сличан

образац понашања током времена. Супротно, уколико су параметри кретања временске серије функције временског тренутка, тада је временска серија нестационарна.

За стационарне временске серије могу тачно да се одреде њихови први и други моменти, као и аутоковаријансна функција, што омогућава лакше и прецизније предвиђање будућих вредности временских серија. Као што је већ поменуто у поглављу 2.1, постоје два вида стационарности. У случају строге стационарности временска серија је стационарна ако се њена својства не мењају транслирањем у времену. То значи да случајне промењљиве које припадају тој серији имају исту очекивану вредност, дисперзију и моменте вишег реда. Ако пак имамо слабу стационарност, тада важи да се очекивана вредност и дисперзија не мењају током времена, док је аутоковаријациона функција између свака два члана серије само функција временског растојања између њих. Нестационарна временска серија се може трансформисати тако да се добије стационарна временска серија о чему ће бити више речи у наставку.

Свака временска серија се може разложити на одређени број компоненти. Постоје четири основне компоненте које се јављају при разлагању временских серија, то су тренд $f(t)$, сезонска $s(t)$, циклична $c(t)$ и случајна компонента $\varepsilon(t)$.

Тренд временске серије представља дугрочну правилност у кретању временске серије и описује њено основно понашање у дужем временском периоду, или прецизније у читавом временском периоду за који је временска серија позната. У зависности да ли вредности временске серије током времена систематски опадају или расту, тренд може бити опадајући или растући. Такође, тренд може бити детерминистички или стохастички, у зависности да ли се промене временске серије током времена могу предвидети, односно да ли се промене могу апроксимирати неком детерминистичком функцијом или се оне дешавају случајно.

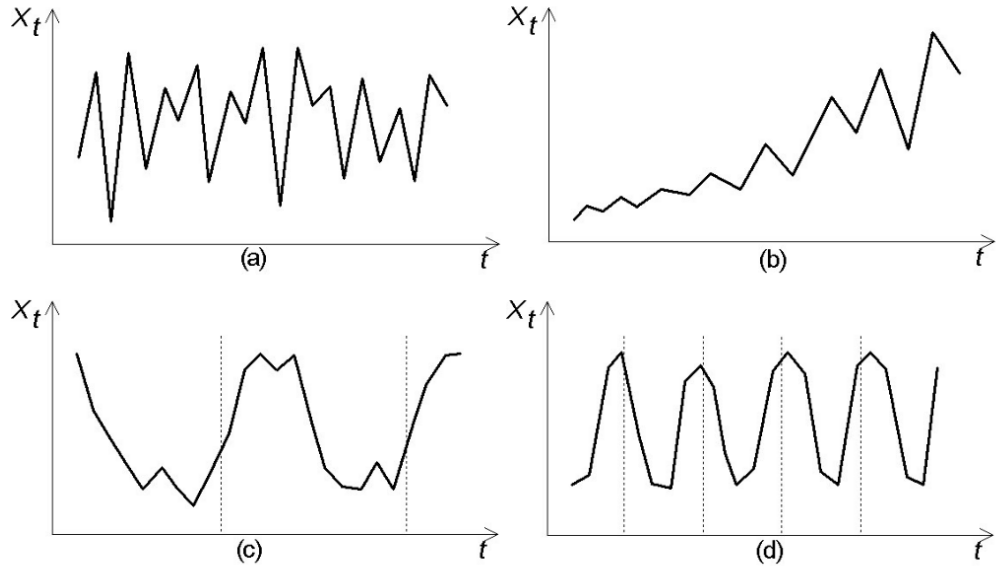
Врло често је неопходно трансформисати нестационарну временску серију у стационарну при формирању модела како би се добили добри резултати. Први, класичан приступ полази од тога да се средина нестационарног процеса може представити детерминистичком функцијом времена. Тада за временску серију кажемо да садржи детерминистички тренд. Други приступ заснован је на поступку диференцирања временске серије. Ако се поступком диференци-

рања добија стационарна временска серија, тада првобитна временска серија садржи стохастички тренд.

Сезонска компонента временске серије представља правилност у кретању временске серије која се појављује периоду до једне календарске године који се назива период сезоне. У практичним применама већина временских серија представља податке прикупљене у узастопним временским интервалима, као што су дани, недеље, месеци, квартали, итд. Због тога поједине временске серије испољавају правилности у свом кретању које се јављају у периодима до годину дана (месечно, квартално или полугодишње) и њих називамо још и сезонским временским серијама. Присуство сезонске компоненте утиче на то да постоји већи степен корелације између података добијених у истој сезони него између оних у различитим сезонама. До сезонских осцилација долази, на пример, због различитих временских услова, односно појава које се могу приписати климатским или календарским факторима. Тако ће, на пример, продаја освежавајућег пића у летњем периоду, бити по правилу знатно виша од продаје у зимском периоду.

Циклична компонента представља правилност у кретању временске серије која се појављује у одређеним, често неједнаким, периодима. На пример, трајање неке моде током извесног броја година и њене замене новом модом током следећих година. У пракси, за компоненту тренда се претпоставља да укључује и цикличну компоненту. Идентификација циклуса и периодичности је један од важних задатака анализе временских серија јер омогућава добру прогнозу будућих вредности временске серије.

Претходне три компоненте се једним именом називају неслучајне компоненте. Ниједна од ових компонената није обавезна у моделу. Једина компонента коју сваки модел мора да има је случајна компонента која служи за објашњавање непредвидивих флуктуација. **Случајна компонента** временске серије представља промене које су случајног карактера, односно флуктуације које не припадају ни сезонским, ни цликличним променама, као ни тренду. Вредности у различитим временским тренуцима међусобно су независне, па се кретање ове компоненте у времену не може предвидети.



Слика 2.1: **Примери временских серија:** (a) временска серија стационарна у односу на средњу вредност, (b) временска серија са узлазним трендом и случајним флуктуацијама око њега (нестационарна у односу на средњу вредност), (c) временска серија са цикличним варијацијама, (d) временска серија са сезонским варијацијама [23].

У случају да временска серија садржи све четири наведене компоненте, постоје три општа модела која могу описивати такву временску серију:

- **Адитивни модел** $X_t = f(t)+s(t)+c(t)+\varepsilon(t)$. Овај модел претпоставља да се све компоненте понашају независно једна од друге.
- **Мултипликативни модел** $X_t = f(t) \cdot s(t) \cdot c(t) \cdot \varepsilon(t)$. Овај модел претпоставља да све компоненте међусобно зависе једна од друге.
- **Мешовити модел** $X_t = f(t) \cdot c(t) \cdot (s(t) + \varepsilon(t) - 1)$. Овај модел представља комбинацију адитивног и мултипликативног модела. У њему се претпоставља да су сезонска и случајна компонента зависне од тренда и цикличне компоненте, али да су оне међусобно независне.

Изравнање временских серија

У великом броју временских серија присутне су флуктуације које отежавају уочавање основних компонената временске серије и њихово описивање. Поступци изравнања временске серије полазе од претпоставке да је у временској серији присутна извесна законитост понашања опсервација заједно са случаним флуктуацијама, а циљ њихове примене је добијање „изравнате" временске серије са пригушеним флуктуацијама. Свођење поменутих флуктуација на што мању меру одвија се тако што се старијим подацима посматране временске серије не додељује једнака важност као новијим подацима.

Претпоставимо да се временска серија X_t може представити моделом у коме посматрана појава има случајне флуктуације око просечног нивоа μ_t. У том случају модел временске серије X_t може се записати као $X_t = \mu_t + \varepsilon_t$ где је $\varepsilon_t = \varepsilon(t)$ случајна компонента за коју важи $E(\varepsilon_t) = 0, D(\varepsilon_t) = \sigma^2 = const.$ Оцена просечног нивоа μ_t може се вршити коришћењем пондерисане средине која већи пондер даје новијим подацима. Коришћењем експоненцијално опадајућих пондера долазимо до $\mu_t = \alpha X_t + \alpha(1-\alpha)X_{t-1} + \alpha(1-\alpha)^2 + ... = \alpha \sum_{j=0}^{\infty} (1-\alpha)^j X_{t-j} = \alpha\left[1 + (1-\alpha)B + (1-\alpha)^2 B^2 + ... + (1-\alpha)^j B^j + ...\right] X_t$. Како израз у средњој загради представља збир чланова опадајуће геометријске прогресије он износи $[1-(1-\alpha)B]^{-1}$, односно претходни израз можемо записати у облику $[1-(1-\alpha)B]\mu_t = \alpha X_t$.

Коначно, добијамо израз $\mu_t = \alpha X_t + (1-\alpha)\mu_{t-1}$ који текућу вредност нивоа μ_t представља као линеарну комбинацију, односно пондерисани просек текуће вредности временске серије X_t и претходне вредности нивоа μ_{t-1}. Пондер α назива се **константа изравнања**, а претходно описани поступак рекурзивног одређивања текуће вредности назива се **једноставно експоненцијално изравнање** [23].

Алтернативни и чешће коришћени начин изражавања једноставног експоненцијалног изрававнања јесте у облику корекције грешком. Према њему се текућа вредност нивоа добија модификацијом претходне вредности нивоа делом текуће грешке $\epsilon_t = X_t - \mu_{t-1}$ која је настала коришћењем оцене нивоа из претходног периода као прогнозе у текућем периоду. Модел у овом облику пишемо као $\mu_t = \mu_{t-1} - \alpha(X_t - \mu_{t-1}) = \mu_{t-1} + \alpha\epsilon_t$.

Једноставно експоненцијално изравнање корисно је за разумевање процеса изравнања и релевантно је у случајевима када се временска серија коју желимо да изравнамо представља константни процес са случајним флуктуацијама. Међутим, уколико желимо да изравнамо временску серију са трендом и сезоналношћу, тада је њен модел нешто компликованији и у случају адитивног модела се може записати као $X_t = \mu_t + f_t + s_t + \varepsilon_t$. Видимо да у моделу поред просечног нивоа μ_t сада фигуришу и тренд $f_t = f(t)$ и сезоналност $s_t = s(t)$ временске серије.

Један од познатијих метода експоненцијалног изравнања временских серија које као компоненте имају тренд и сезоналност јесте **Холт-Винтерсов**[1] **метод експоненцијалног изравнања** који би у облику корекције грешком за адитивни модел био дефинисан следећим једначинама:

- $\mu_t = \mu_{t-1} + T_{t-1} + \alpha\epsilon_t$
- $f_t = f_{t-1} + \alpha\gamma\epsilon_t$
- $s_t = s_{t-s} + \delta(1-\alpha)\epsilon_t$

где су α, γ и δ константе изравнања, а s период сезоне.

2.4 Модели временских серија

Један од начина за изучавање појава које се могу посматрати као временске серије је изградња модела за такве серије. Основна мотивација за изградњу модела најчешће је жеља да се прогнозирају будуће вредности временске серије. У наставку ће бити описани неки од основних модела временских серија.

Бели шум

Случајна компонента поменута у претходном поглављу познатија је као бели шум. Бели шум је слабо стационаран процес. Иако се о белом шуму ретко говори у пракси, он је веома битан јер су многи комплекснији стационарни процеси изграђени на основу њега [23].

[1] Holt-Winters

Дефиниција 2.4.1 *Процес* $\{\varepsilon_t, t \in T\}$ *назива се* ***бели шум*** *ако су* $\varepsilon_1, \varepsilon_2$ *некорелисане случајне величине, нулте средње вредности и стабилне дисперзије, односно ако важи:*

- $E(\varepsilon_t) = \mu_\varepsilon = 0, \forall t \in \mathbf{Z}$;
- $D(\varepsilon_t) = E(\varepsilon_t^2) = \sigma^2 = const, \forall t \in \mathbf{Z}$;
- $cov(\varepsilon_t, \varepsilon_{t-k}) = E(\varepsilon_t \varepsilon_{t-k}) = 0, \forall t \in T, k = 1, 2, .., k \neq 0$.

Линеарни процеси

Линеаран модел представља најопштији оквир за моделовање стационарних временских серија. Једна од најчешћих претпоставки јесте да је веза између улазних и излазних параметара линеарна [24, 20, 22].

Према Волдовој[2] теореми имамо да се сваки слабо стационарни процес може представити као збир два међусобно некорелисана процеса, једног чисто детерминистичког и једног чисто недетерминистичког.

Дефиниција 2.4.2 *Процес* $\{X_t, t \in T\}$ *је* ***линеаран*** *ако се може представити у облику* $X_t = \mu_t + \sum_{j=0}^{\infty} \psi_j \varepsilon_{t-j}$, *где је* μ_t *детерминистичка компонента,* $\{\varepsilon_t, t \in T\}$ *процес белог шума, а* $\psi_1, \psi_2, ...$ *непознати параметри.*

За линеарне процесе важи:

- $E(X_t) = \mu_t$;
- $D(X_t) = \sigma^2 \sum_{j=0}^{\infty} \psi_j^2$;
- $\gamma_k = \sigma^2 \sum_{j=0}^{\infty} \psi_{j+k} \psi_j$;
- $\rho_k = \frac{\gamma_k}{\gamma_0} = \frac{\sum_{j=0}^{\infty} \psi_{j+k} \psi_j}{\sum_{j=0}^{\infty} \psi_j^2}$.

[2]Herman Ole Andreas Wold, шведски математичар

На основу линеарног процеса могу се извести три класе процеса са коначним бројем параметара којима описујемо слабо стационарне процесе:

- ауторегресиони модел реда p, AR(p);
- модел покретних просека реда q, MA(q);
- ауторегресиони модел покретних просека ARMA(p, q), који је комбинација AR(p) и MA(q) процеса.

Ауторегресиони модел реда p, AR(p)

Ауторегресиони модели, као што им име каже, имплицирају регресију на претходне вредности посматране временске серије. Ови модели имају природну интерпретацију где се наредна опсервација може дефинисати као блага промена вредности добијене применом једноставне функције над најскоријим опсервацијама [23].

Дефиниција 2.4.3 *$\{X_t, t \in T\}$ је **ауторегресиони модел реда** p ако се може представити у облику $X_t = \mu + \sum_{j=1}^{p} \phi_j X_{t-j} + \varepsilon_t$, где је μ константа, $\{\varepsilon_t, t \in T\}$ бели шум, а $\phi_j, j = 1, ..., p$ непознати параметри.*

Дефиниција 2.4.4 ***Оператор кашњења*** *B је алгебарска величина која се дефинише као $BX_t = X_{t-1}$, а која текућу опсервацију временске серије помера један период у назад (у прошлост). У општем случају имамо да важи $B^0 X_t = X_t, B^k Xt = X_{t-k}$ и $k \in \mathbf{Z}$.*

С обзиром на то да је μ константа, она се може изоставити у запису AR(p) модела, тада имамо $X_t = \sum_{j=0}^{p} \phi_j X_{t-j} + \varepsilon_t = \phi_1 X_{t-1} + ... + \phi_p X_{t-p} + \varepsilon_t$. Како имамо да је текућа вредност процеса линеарна комбинација p прошлих вредности посматране временске серије и белог шума претходни израз се може лепше записати коришћењем оператора кашњења $\phi(B) X_t = \varepsilon_t$, где је $\phi(B) = 1 - \phi_1 B - \phi_2 B^2 - ... - \phi_p B^p$.

Једначина $\phi(z) = 0, z \in \mathbf{C}$ је карактеристична једначина AR(p) модела, а њена решења су карактеристични корени AR(p) модела. Уколико су сви њени корени по модулу строго мањи од један, тада је временска серија стационарна. Уколико постоји бар један корен који је по модулу једнак један, тада је временска серија нестационарна, док уколико постоји бар један корен који је по модулу већи од један за временску серију кажемо да је експлозивна.

Модел покретних просека реда q, MA(q)

Модел покретних просека је користан у моделовању појава код којих догађаји узрокују тренутне ефекте, а који трају кратак период времена. Овај модел се доводи у везу са поступком изравнања временске серије. Методе изравнања које су базиране на аритметичкој средини називају се методе изравнања помоћу покретних просека. Термин покретни просек је узет из разлога што се сваким формирањем нове вредности серије мења стара и рачуна нова вредност аритметичке средине, тако да се као резултат, средња вредност, тј. просек, увек мења, тј. покреће [23].

Нека је дат низ података $X_1, X_2, ..., X_{t-m}, ..., X_{t-1}, X_t, X_{t+1}, ..., X_{t+m}, ..., X_T$. Сваки податак X_t замењује се новим податком $Y_t = \sum_{i=-m}^{m} a_i X_{t-i}$, за које важи $t = m+1, ..., T-m$, $\sum_{i=-m}^{m} a_i = 1$. Наведени поступак може се применити више пута, чиме се губи m првих и m последњих података.

Дефиниција 2.4.5 *$\{X_t, t \in T\}$ је **модел покретних просека реда** q ако се може представити у облику $X_t = \varepsilon_t - \sum_{j=1}^{q} \theta_j \varepsilon_{t-j} = \varepsilon_t - \theta_1 \varepsilon_{t-1} - ... - \theta_q \varepsilon_{t-q}$. где $\{\varepsilon_t, t \in T\}$ бели шум, а $\theta_j, j = 1, ..., p$ непознати параметри. Коришћењем оператора кашњења, претходни израз се може записати као $X_t = \theta(B)\,\varepsilon_t$ где је $\theta(B) = 1 - \theta_1 B - \theta_2 B^2 - ... - \theta_q B^q$.*

Једначина $\theta(z) = 0, z \in \mathbf{C}$ је карактеристична једначина MA(q) модела, а њена решења су карактеристични корени MA(q) модела. Уколико су сви корени карактеристичне једначине по модулу већи од јединице тада кажемо да је временска серија описана моделом покретних просека инвертибилна.

Нека су дати AR(p), $\varepsilon_t = \phi(B)\,X_t$ и MA(q), $X_t = \theta(B)\,\varepsilon_t$ модели. Тада важи $X_t = \theta(B)\,\varepsilon_t = \theta(B)\,\phi(B)\,X_t \Rightarrow \theta(B)\,\phi(B) = 1 \Rightarrow \theta(B) = \phi(B)^{-1}$ што даје везу између коефицијената ових модела.

На основу претходног, видимо да стационарну временску серију можемо описати или ауторегресионим моделом покретних просека коначног реда или моделом покретних просека бесконачног реда. Такође, инвертибилну временску серију можемо описати или ауторегресионим моделом бесконачног реда или моделом покретних просека коначног реда.

Ауторегресиони модел покретних просека ARMA(p, q)

Проблем са претходно описаним репрезентацијама AR и MA модела је тај што оне могу да садрже превелики број параметара, чак и у случају модела коначног реда, јер су за добру апроксимацију често потребни модели вишег реда. Велики број параметара умањује ефикасност оцењивања, због чега у моделовању може бити неопходно укључити и ауторегресионе компоненте и компоненте покретних просека, што нас доводи до ауторегресионог процеса покретних просека ARMA [10].

Дефиниција 2.4.6 $\{X_t, t \in T\}$ *је* ***ауторегресиони модел покретних просека*** $ARMA(p,q)$ *ако се може представити у облику* $X_t = \sum_{j=1}^{p} \phi_j X_{t-j} + \varepsilon_t - \sum_{i=1}^{q} \theta_i \varepsilon_{t-i}$ *или у еквивалентној форми* $\phi(B) X_t = \theta(B) \varepsilon_t$ *где је* $\{\varepsilon_t, t \in T\}$ *бели шум,* $\phi(B) = 1 - \phi_1 B - \phi_2 B^2 - ... - \phi_p B^p$ *AR карактеристични полином ARMA(p,q) модела,* $\theta(B) = 1 - \theta_1 B - \theta_2 B^2 - ... - \theta_q B^q$ *MA карактеристични полином ARMA(p,q) модела.*

У ARMA(p, q) моделу, p представља ред ауторегресионе компоненте, а q ред компоненте покретних просека. Другачије се ARMA(p, q) процес може записати и као $X_t - \sum_{j=1}^{p} \phi_j X_{t-j} = \varepsilon_t - \sum_{i=1}^{q} \theta_i \varepsilon_{t-i}$, где лева страна једнакости представља AR компоненту, а десна страна једнакости представља MA компоненту. Специјални случајеви ARMA(p, q) модела су AR(p) = ARMA(p, 0) и MA(q) = ARMA(0, q).

Интегрисани ауторегресиони модел покретних просека ARIMA(p,q,d)

Често се у раду са ARMA моделима јавља проблем стационарности. Овај проблем се превазилази увођењем ARIMA модела [21]. ARIMA модел нестационарно временску серију прво своди на стационарну диференцирањем, а затим је моделује ARMA моделом [22].

Дефиниција 2.4.7 *Операција одузимања узастопних опсервација* $\Delta X_t = X_t - X_{t-1} = (1 - B) X_t$ *назива се* ***диференцирање****, а оператор* Δ *је диференцни оператор. Диференцни оператор се може изразити и коришћењем оператора кашњења* $\Delta = 1 - B$*, где важи и* $\Delta^k = (1 - B)^k$*.*

Дефиниција 2.4.8 *Случајан процес* $\{X_t, t \in T\}$ *је* ***интегрисан*** *процес реда* d *у ознаци* $I(d)$ *ако може бити трансформисан у стационаран случајан процес диференцирањем* d *пута.*

Дефиниција 2.4.9 *ARIMA(p, d, q) (енгл.* autoregressive integrated moving average*) модел дат је изразом* $\phi_p(B)(1-B)^d X_t = \theta_0 + \theta_q(B)\varepsilon_t$*, при чему AR компонента* $\phi_p(B) = 1 - \phi_1 B - \phi_2 B^2 - ... - \phi_p B^p$ *реда* p *и MA компонента* $\theta_q(B) = 1 - \theta_1 B - \theta_2 B^2 - ... - \theta_q B^q$ *реда* q *немају заједничких фактора, док* d *означава ниво интегрисаности временске серије.*

За разлику од ARMA(p, q) модела, код ARIMA(p, d, q) модела уведена је и константа θ_0, која има различите улоге у зависности од реда диференцирања, односно од вредности d. Ако је $d = 0$, временска серија је стационарна, а константа је у релацији са средином процеса $\theta_0 = \mu(1 - \phi_1 - ... - \phi_p)$, где је $\mu = E(X_t)$. За вредности $d > 0$, θ_0 указује на присуство детерминистичког тренда у серији. Када нема места претпоставци да ће диференцирана серија садржати детерминистички тренд константа θ_0 се може изоставити из израза.

Неки специјални случајеви ARIMA(p,d,q) модела су AR(p) = ARMA(p, 0, 0), MA(q) = ARMA(0, 0, q), бели шум = ARIMA(0, 0, 0), случајни ход = ARIMA(0, 1, 0).

Сезонски ARIMA модели

Као што је поменуто раније, временске серије које имају изражену сезонску компоненту називају се сезонске временске серије. При креирању модела за временску серију са израженом сезонском компонентом изостављање сезонског фактора довело би до формирања модела који не би био оптималан за дату временску серију. Идеја је креирање економичног ARIMA модела који ће са релативно малим бројем додатних коефицијената у односу на несезонске моделе успешно моделовати и сезонске промене временске серије [23]..

Дефиниција 2.4.10 *Најмањи временски период у коме се понови уочена појава назива се* ***период сезоне*** *и означава се са* s*.*

Ако су у питању месечне серије, тада је период сезоне $s = 12$, за кварталне сезоне $s = 4$, а за полугодишње $s = 2$. С обзиром да на то да се посматрана појава понавља са извесним правилностима после периода сезоне, очекује се да ће опсервације раздвојене периодом бити међусобно корелисане.

Постепено смањивање вредности аутокорелационих коефицијената на сезонским кашњењима $(s, 2s, 3s, ...)$ показатељ је сезонске нестационарности. Слично поступку елиминисања нестационарности код несезонских временских серија коришћењем оператора диференцирања $(1 - B)$, за отклањање сезонске нестационарности користимо оператор сезонског диференцирања $(1 - B^s)$.

Дефиниција 2.4.11 $\{X_t, t \in T\}$ *је* ***сезонски ARIMA модел*** *ако се може представити у облику* $\phi_p(B)(1 - B)^d(1 - B^s)^D X_t = \theta_q(B)\varepsilon_t$, *где су оператори несезонског и сезонског диференцирања примењени d, односно D пута, респективно.*

При изградњи овако дефинисаног сезонског ARIMA модела постоје извесне потешкоће, као што је услов да минимално један полином ($\phi_p(B)$ или $\theta_q(B)$) мора бити реда s, због чега су Бокс и Џенкинс приступили дефинисању класе такозваних мултипликативних сезонских ARIMA модела.

Претпоставка је да је у општем случају временска серија X_t, за коју је утврђено да има карактеристике сезонске временске серије, моделована коришћењем несезонског ARIMA модела $(1 - B)^d \phi_p(B) X_t = \theta_q(B)\nu_t$, где су $\phi_p(B) = 1 - \phi_1 B - \phi_2 B^2 - ... - \phi_p B^p$ и $\theta_q(B) = 1 - \theta_1 B - \theta_2 B^2 - ... - \theta_q B^q$ полиноми по операторе кашњења, реда p и q, респективно. Због сезонског карактера серије процес ν_t неће бити процес белог шума. Коефицијенти аутокорелације овог процеса на сезонским кашњењима биће различити од нуле, због чега се и овај процес моделује ARIMA моделом $(1 - B^s)^D \Phi_P(B^s)\nu_t = \Theta_Q(B^s)\varepsilon_t$, где су $\Phi_P(B^s) = 1 - \Phi_1 B^s - \Phi_2 B^{2s} - ... - \Phi_P B^{Ps}$ и $\Theta_Q(B^s) = 1 - \Theta_1 B^s - \Theta_2 B^{2s} - ... - \Theta_Q B^{Qs}$ полиноми по оператору B^s, реда P и Q респективно.

Комбиновањем претходних израза добијамо **Бокс-Џенкинсов мултипликативни сезонски ARIMA модел** $(1 - B)^d (1 - B^s)^D \phi_p(B) \Phi_P(B^s) X_t = \theta_q(B) \Theta_Q(B^s) \varepsilon_t$.

Уобичајено је да се у овом моделу полиноми $\phi_p(B)$ и $\theta_q(B)$ називају регуларним ауторегресионим полиномом и полиномом покретних просека, а полиноми $\Phi_P(B^s)$ и $\Theta_Q(B^s)$ сезонским ауторегресионим полиномом и полиномом покретног просека. Стандардна ознака мултипликативних сезонских ARIMA модела са периодом сезоне s је $ARIMA(p,d,q) \times (P,D,Q)_s$.

2.5 Изградња ARIMA модела

Поступак моделовања ARIMA процеса дефинисали су Бокс (енгл. *George Box*) и Ценкинс (енгл. *Gwilym Jenkins*), па се према њима он назива Бокс-Ценкинсонов поступак. Квалитетан модел треба да прође кроз три етапе ове методологије: идентификацију модела, оцењивање и проверу адекватности, као и да задовољи основне принципе који карактеришу добар модел [23]:

- **Економичност** – Циљ је описати појаву што једноставнијим моделом (који садржи релативно мали број коефицијената) који ће истаћи суштинску карактеристику изучаване појаве.
- **Идентификабилност** – Представља принцип без кога није могуће оцењени модел интрепретирати на задовољавајући начин. Без идентификације модела постоје бар два скупа вредности коефицијената који су сагласни са подацима.
- **Конзистентност са подацима** – Провера конзистентности модела са подацима спроводи се коришћењем различитих тестова провере адекватности модела. Од модела се очекује да обезбеди добро прилагођавање подацима, док резидуали треба да буду релативно мали и да имају карактеристике потпуно случајног процеса
- **Конзистентност са теоријом** – Модел мора бити конзистентан са априорним знањем, без обзира на то да ли је извор тог знања у научној теорији или здравом разуму. Теорија може сугерисати величину или предзнак коефицијената, а оцењени модел треба да буде сагласан са том информацијом.
- **Прихватљивост података** – Модел не сме да предвиђа вредности које не задовољавају нека дефинисана ограничења (нпр. модел не сме давати негативне вредности за бруто друштвени прозвод).

- **Успешност прогнозирања** – Указује на прецизност прогноза добијених на основу модела временских серија. У блиској је вези са појмом структурне стабилности, односно константности коефицијената модела. Критеријум успешности прогнозирања проверава се тако што се користе опсервације ван узорка за оцењивање у циљу провере степена прецизности прогнозе модела. Уколико један од модела има мању средње квадратну грешку прогнозе, а све остале карактеристике модела су једнаке, тада за тај модел кажемо да је успешнији са становишта прогнозирања, па је прихватљивији за коришћење од конкурентних модела.

- **Обухватност** – За модел се каже да обухвата конкурентни модел ако може да објасни резултате добијене тим моделом. У том случају конкурентни модел не садржи информације које би се могле користити у циљу побољшања изабраног модела. Модел треба не само да објасни, односно опише податке, већ и да објасни успех или промашај конкурентног модела у објашњењу истих података.

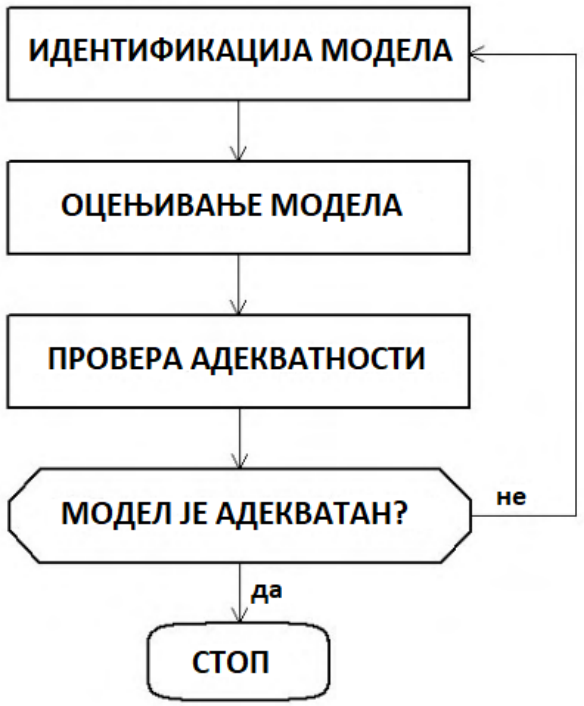
Основу Бокс-Џенкинсовог поступка чине три етапе изградње:

- **Идентификација** – Под етапом идентификације подразумева се поступак коришћења података временске серије у циљу издвајања уже класе економичних ARIMA модела, који се узимају у разматрање као потенцијални генератори датог скупа података. На основу графика и корелограма најпре се утврди потреба за трансформацијама, затим се одреди одговарајућа класа AR, MA или ARMA модела из које се бира одговорајући модел са којим се улази у наредну етапу.

- **Оцењивање** – Етапа оцењивања модела представља поступак закључивања о коефицијентима модела на основу расположивих података, што је условљено адекватношћу изабраног модела. Методи оцењивања ARIMA модела који су засновани на нумеричким поступцима максимизације функције критеријума захтевају познавање почетних оцена коефицијената. Коришћењем ових почетних оцена започиње се итеративни процес долажења до коначних оцена коришћењем методе нелинеарних најмањих квадрата или највеће веродостојности. Након оцене изабраног модела прелазимо на наредну етапу.

- **Провера адекватности** – Подразумева поступак суочавања прилагођеног модела подацима у циљу откривања његових евентуалних недостатака. Основни кораци у оквиру етапе провере адекватности модела подразумевају проверу статистичке значајности оцењених коефицијената и испуњености претпоставке да резидуали оцењеног модела представљају процес белог шума. Уколико се покаже да модел има недостатака, могуће је његово побољшање и тада се наставља поступак градње ARIMA модела за дату временску серију. У супротном, адекватан модел се може користити у сврхе прогнозирања.

На слици 2.2 можемо видети графички приказ дијаграма тока Бокс-Ценкинсоновог поступка.



Слика 2.2: Дијаграм тока Бокс-Ценкинсоновог поступка.

Глава 3

Рачунарство у облаку

Идеја да се рачунарски ресурси – капацитети за рачунање, складиштење и пренос података – испоручују слично комуналним услугама, као што су снабдевање електричном енергијом и водом или јавни превоз, стара је више од пола века [15]. Ову идеју први је јавно изнео Џон Макарти (енгл. *John McCarthy*) 1961. године у говору на прослави поводом стогодишњице MIT-а (енгл. *Massachusetts Institute of Technology*).

3.1 Шта је рачунарство у облаку?

Једноставно речено, рачунарство у облаку (енгл. *cloud computing*) је испорука рачунарских услуга – укључујући сервере, складиште, базе података, оперативни систем, аналитички софтвер и друге хардверске и софтверске ресурсе – преко интернета како би се омогућио бржи развој, флексибилност ресурса и њихово лакше скалирање.

Рачунарство у облаку се све више развија због потреба компанија да смање трошкове приликом куповине хардвера и софтвера. Са овим типом рачунарства избегавају се велика почетна улагања, трошење много времена на управљање хардвером и креирање сопствене инфраструктуре, као и размишљање о лиценцирању потребног софтвера. Уместо тога, потребно је само дефинисати који тачно тип и величина рачунарских ресурса су неопходни за пословање и њима се може се приступити скоро тренутно, без обзира на количину која је неопходна.

Основу рачунарства у облаку чини конвергентна инфраструктура, коју чине различите технологије повезане у једну логичку и функционалну це-

лину, апстракција физичких ресурса виртуелизацијом и дељење ресурса. У моделу рачунарског облака разликујемо два дела система: спољни (енгл. *front end*), који је кориснички део и обухвата оне делове инфраструктуре који су под контролом корисника као и сам начин приступа корисника услузи и унутрашњи (енгл. *back end*), који обухвата инфраструктуру пружаоца услуга облака. Овај модел дозвољава организацијама да подигну и користе апликације много брже, са бољом контролом и мање одржавања, што омогућава брже и ефикасније испуњавање променљивих и непредвидивих захтева пословања.

Како бисмо боље разумели рачунарство у облаку размотримо следећи пример. Замислимо да имамо директора неке велике корпорације. Његове обавезе укључују доношење одлука о куповини сигурног и поузданог софтвера и хардвера како би његови запослени могли да несметано раде свој посао. У сваком тренутку потребно је обезбедити довољно рачунарских ресурса за несметан рад, али потреба за ресурсима често варира током времена, приликом захтевних операција потреба за ресурсима достиже максимум, док већину времена искоришћеност ресурса није оптимална. Код рачунарства у облаку, уместо обезбеђивања свог неопходног хардвера и софтвера, користи се веб апликација која омогућава запосленима приступ свим ресурсима потребним за рад. Где год да се корисник налази, најчешће му је довољан само веб прегледач за приступ свим ресурсима у облаку, чиме су хардверски и софтверски захтеви сведени на минимум. У овом случају локални рачунари више не морају да обављају захтевне операције, потребна је само добра интернет веза, а све операције се извршавају на мрежи рачунара које чине облак. Због свега овога компаније све чешће одлучују да користе рачунарство у облаку и изнајмљују ресурсе који су у власништу других компанија.

Постоје бројне дефиниције рачунарског облака, али обично се користи дефиниција коју је дао Амерички институт за стандарде (енгл. *National Institute of Standards and Technology, NIST*) [16]: „Рачунарски облак је модел окружења које омогућава свуда присутан, погодан мрежни приступ дељивим рачунарским ресурсима (мрежним, серверима, складишту података, апликацијама и сервисима), који на захтев корисника и уз минималну интеракцију са пружаоцем услуга могу брзо бити стављени на располагање кориснику или отказани."

3.2 Карактеристике рачунарства у облаку

Неке од стандардних карактеристика рачунарства у облаку су [1, 4]:

- **Пружање услуге на захтев корисника** – Корисник има могућност да самостално бира и покреће изабране рачунарске ресурсе. Услуге рачунарства у облаку се пружају самоуслужно и на захтев, тако да се велике количине рачунарских ресурса могу обезбедити за неколико минута, дајући корисницима велику флексибилност и ослобађајући их од притиска планирања капацитета.

- **Независност уређаја и локације** – Корисницима је омогућено да свим услугама у облаку приступе преко веб прегледача без обзира на њихову тренутну локацију или тип уређаја који користе. Конфигурација уређаја са кога се приступа облаку није битна и подржани су уређаји свих типова са различитим оперативним системима. Најчешће је једини предуслов за рад у облаку добра интернет конекција.

- **Удруживање ресурса** – Ресурси инфраструктуре пружаоца услуга се удружују и могу опслужити више корисника комбинујући различите физичке и виртуелне ресурсе који се динамички додељују према захтевима корисника.

- **Брза еластичност** – Услуге у облак окружењу могу бити брзо покренуте у циљу повећања или смањења потребних ресурса на основу захтева корисника, промена ресурса дешава се у периоду од пар минута. У зависности од подешавања, неке од функционалности се могу и аутоматски скалирати како би се оптимизовала искоришћеност на основу одговарјућих метрика које се могу пратити.

- **Смањење трошкова** – Инфраструктура је обезбеђена од стране трећег лица што смањује трошкове куповине софтвера и хардвера, поготово оног који би се користио само за једнократне или повремене послове. Такође, како је инфраструктура обезбеђена од стране трећег лица нема потребе за утрошеним временом и неопходним вештинама неопходним за њену имплементацију, одржавање и унапређивање. Плаћање се врши само за оне ресурсе које корисник користи и за оно време током којих су ти ресурси коришћени.

- **Поузданост** – Услуге рачунарства у окружењу облака подразумевају резервне копије података, опоравак од катастрофе и континуитет пословања. Подаци се обично налазе на више редудантних локација на мрежи пружаоца услуга, а у случају да до проблема дође, најчешће на располагању корисник има тимове за подршку који се баве отклањањем насталих проблема. Такође, услуге рачунарства у облаку раде на светској мрежи центара података, који се редовно надограђују на најновију генерацију хардвера и софтвера.

3.3 Врсте облака

Нису сви облаци исти и не постоји универзални тип облака који би био прави за све његове употребе. Рецимо, неким корисницима је битније смањење трошкова, док је другима битнија већа поузданост или безбедност. Због тога постоји неколико различитих врста облака, као и услуга које оне нуде, како би се задовољиле специфичне потребе различитих корисника. Постоје три основне врсте облака: јавни облак, приватни облак и хибридни облак.

Јавни облак

Јавни облаци су најчешћа врста примене рачунарства у окружењу облака. Ресурси у облаку, као што су сервери и складиште, су у власништву независног пружаоца услуга који њима управља и испоручују се преко интернета. Сав хардвер, софтвер и друга пратећа инфраструктура, су у власништву пружаоца услуга и он ради на њиховом одржавању и унапређењу, односно нису доступни корисницима на увид или контролу [12].

У јавном облаку иста инфраструктура, хардвер, складиште и мрежни уређаји, дели се између више организација, односно „закупаца” у облаку, који приступају услугама и управљају својим налогом помоћу веб прегледача. Неке од предности јавног облака су:

- нижи трошкови – нема потребе за куповином хардвера или софтвера и плаћају се само оне услуге које се користе;
- нема потребе за одржавањем – одржавање обезбеђује пружалац услуга;

- веома добра скалабилност – сви потребни ресурси су доступни на захтев како би задовољили различите пословне потребе корисника;
- поузданост – велика мрежа сервера штити од квара и губитка података.

Приватни облак

Приватни облак се састоји од ресурса у облаку које користи искључиво један корисник, једно предузеће или организација. Ова врста облака може бити физички лоцирана у центру података који се налази у оквиру саме организације или може бити хостован од стране независног пружаоца услуге. У приватном облаку, услуге и инфраструктура се одржавају на приватној мрежи, а хардвер и софтвер који се користи је додељен искључиво једној организацији [12].

На овај начин, приватни обак може олакшати организацији да прилагоди своје ресурсе како би задовољила специфичне захтеве које на најбољи начин. Приватне облаке често користе владине агенције, финансијске институције и друге организације са критичним операцијама које захтевају већи ниво контроле над окружењем у коме раде. Предности ове врсте облака су:

- већа флексибилност – могуће је прилагодити облак окружење како би се задовољиле специфичне пословне потребе једног корисника;
- већи ниво контоле – ресурси се не деле са другима, тако да је омогућен већи ниво контроле и већа приватност.

Хибридни облак

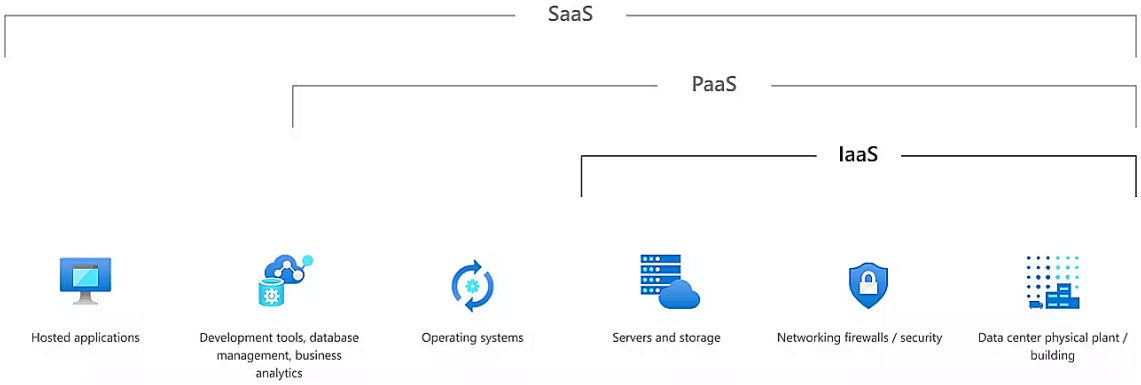
Хибридни облаци комбинују јавне и приватне облаке повезане технологијама које омогућавају да се подаци и апликације деле између њих. Дозвољавајући подацима и апликацијама да се крећу између јавних и приватних облака, ова врста облака даје организацијама још већу флексибилност, више опција за њихову примену и помаже у оптимизацији постојеће инфраструктуре, безбедности и усклађености [12].

Када потражња често варира, хибридни облаци дају организацијама могућност да неприметно повећају своју локалну инфраструктуру, коришћењем јавних облака, како би се изборили са повећаним обимом посла уз дељење

само одређеног скупа података са трећом страном. Организације које користе ову врсту облака добијају могућност да користе све предности које пружа јавни облак, док истовремено чувају веома осетљиве податке у сопственом центру података. Коришћење ове врсте облака елиминише потребу за великим трошковима како би се организација носила са краткорочним скоковима потражње, као и у случајевима да је потребно да организација ослободи неке локалне ресурсе за осетљивије податке или апликације. Како се у окружењу облака плаћање врши само за оне ресурсе који се тренутно користе, у случају повременог коришћења организације избегавају трошкове куповине и одржавања додатних ресурса и опреме који могу бити неактивни током дужег временског периода коришћењем јавног облака у власништву независног пружаоца услуга.

3.4 Модели пружања услуга у облаку

Рачунарство у облаку користи модел испоруке услуга познат под акронимом SPI (енгл. *Software, Platform, Infrastructure*) који означава три највеће групе услуга које се пружају у облак окружењу, а то су: инфраструктура као услуга (енгл. *IaaS, Infrastructure as a Service*), платформа као услуга (енгл. *PaaS, Platform as a Service*) и софтвер као услуга (енгл *SaaS, Software as a Service*). Ове услуге се понекад називају „стеком" рачунарства у облаку јер се надовезују једна на другу.



Слика 3.1: Модели пружања услуга у облаку [4].

Инфраструктура као услуга

Инфраструктура као услуга је најосновнија услуга рачунарства у облаку и односи се на праксу коришћења инфраструктуре на бази виртуелних или физичких ресурса на сличан начин ка који се користе комуналне услуге. Користећи ову услугу корисници изнајмљују инфраструктуру – сервере, виртуелне машине, складиште, мреже – од пружаоца услуга у облаку по принципу плаћања само онога што се тренутно користи. Коришћење инфраструктуре као услуге помаже смањењу трошкова одржавања локалних центара података и куповине хардвера. Такође, корисници добијају флексибилност да повећавају и смањују количину ресурса које користе у складу са тренутном потражњом, као и да повећају поузданост своје основне инфраструктуре. Сваки ресурс се нуди као посебна компонента услуге, а плаћање се врши само за оно време када је одређени ресурс био коришћен. Пружалац услуге управља инфраструктуром, док корисници купују, инсталирају, конфигуришу и управљају сопственим софтвером, укључујући оперативне системе и други неопходан софтвер и апликације. Корисничку инфраструктуру прави сам корисник повезивањем сервиса и инсталацијом неопходних апликација, а он је одговоран и за одржавање тог софтвера [12].

Овај тип услуге погодан је за нове компаније које немају велики почетни капитал који би инвестирали у сопствену инфраструктуру, за случајеве где пословање корисника убрзано расте или када је потребно привремено проширење инфраструктуре. Инфраструктура као услуга је најсличнија традиционалном приступу рачунарству, сем што је поступак обезбеђивања ресурса веома кратак, грануларност ресурса веома добра, а корисник је поштеђен свих активности набавке, инсталације и интеграције физичких компонената инфраструктуре.

Платформа као услуга

Платформа као услуга се односи на услуге рачунарства у облаку које обезбеђују окружење за развој, тестирање, испоруку и управљање софтверским апликацијама. Дизајнирана је да олакша програмерима да брзо креирају апликације без бриге о постављању и управљању физичком инфраструктуром, оперативним системом и разјвојним окружењем и подржи комплетан животни циклус веб апликације – развој, тестирање, имплементацију, упра-

вљање и коришћење. Коришћењем ове услуге корисник добија комплетно окружење за рад и развој у облаку, инфраструктуру – сервере, складиште и мрежну инфраструктуру – али и оперативне системе, развојне алате, услуге пословне интелигенције, системе за управљање базама података и слично. Корисник контролише апликације које се покрећу у окружењу облака, има одређена права коришћења, али нема потпуна права над оперативним системом, мрежом или хардвером који те апликације користе [12].

Овај тип услуге нуди све предности које нуди и инфраструктура као услуга уз додатне функције у виду развојних алата и програмских језика што га чини веома погодним за развој апликативног софтвера. Пружаоци услуга имају власништво над платформом и инфраструктуром и одговорни су за њихово одржавање и квалитет услуге дефинисаних у одговарајућем уговору о услузи, док је корисник власник само апликација које он развија. Неки пружаоци услуга дају опције и за развој на више платформи, као што су рачунари, мобилни уређаји и претраживачи, што чини развој апликација на више платформа лакшим и бржим. Такође, платформа као услуга омогућава коришћење софистицираног лиценцираног софтвера и алата за пословну аналитику по повољнијим ценама у односу на цене куповине лиценце за сваки од неопходних софтвера одвојено.

Софтвер као услуга

Софтвер као услуга омогућава испоруку софтверских апликација преко интернета. Пружаоци услуга хостују и управљају апликацијом, платформом и инфраструктуром и одговорни су за сва одржавања, попут надоградње софтвера и безбедносних закрпа [12]. Такође, пружалац услуга је власник и позадинске инфраструктуре, укључујући мрежу, сервисе, оперативне системе, као и конкретног софтвера који је доступан великом броју корисника преко веб претраживача. Корисници изнајмљују употребу апликације и могу се повезивати на апликацију преко интернета, обично преко веб претраживача, на телефону, таблету или рачунару. Уобичајени примери су е-пошта, календар и канцеларијски алати.

Код овог типа услуге пружалац услуга управља и хардвером и софтвером уз одговарајући уговор о услузи који каже да ће пружалац услуге обезбедити доступност апликације и безбедност корисничких података приликом употребе апликације. Корисник овиме смањује трошкове који би настали

куповином лиценци, инсталирањем апликације и обезбеђивањем хардверских ресурса неопходних за њено покретање.

3.5 Архитектура облака

Архитектура облака се односи на структуру и организацију компонената и сервиса који чине облак. Она дефинише начин на који се ресурси, подаци и апликације организују и повезују како би се пружиле услуге рачунарства у облаку. Обично аритектура облака има слојевиту структуру која се састоји од више слојева услуга – инфраструктурних (као што су складишта података и виртуелних машина), платформских (као што су развојна окружења и базе података) и софтверских (апликације које су доступне корисницима).

У архитектури облака виртуелизација има кључну улогу. Виртуелизација омогућава дељење физичких ресурса рачунара, као што су процесорска снага, меморија, складиште, и креирање виртуелних ресурса над њима који се користе за пружање услуга рачунарства у облаку. Овај приступ омогућава ефикасније коришћење хардверских ресурса, већу флексибилност и скалабилност, као и лакше управљање ресурсима.

Кључни елемент виртуелизације представљају виртуелне машине. Виртуелне машине представљају виртуелне рачунарске системе који се извршавају на једном физичком серверу, који се још назива и хост рачунар. Свака виртуелна машина има свој оперативни систем, апликације и ресурсе, што омогућава изолацију и независност виртуелних машина чак и када се извршавају на истом хост рачунару, што значи да се ресурсима и подацима једне виртуелне машине не може приступити од стране других виртуелних машина. Виртуелним машинама на хост рачунару управља софтвер који се назива хипервизор или монитор виртуелних машина. Он је одговоран за креирање, контролисање и рад виртуелних машина, као и пружање изолације између њих. Хипервизор омогућава дељење физичких ресурса између виртуелних машина и обезбеђује њихов несметан упоредни рад. Виртуализацијом је омогућена лакша миграција виртуелних машина између хост рачунара без прекида рада апликације, што омогућава динамичко управњање ресурсима.

Архитектура облака се најчешће ослања на дистрибуиране системе и широку географску расподелу ресурса како би се постигла висока доступност и отпорност на кварове. Виртуелни ресурси се налазе на више физичких

сервера, а врло често и у више различитих складишта података на различитим локацијама, чиме се обезбеђује редундантност и отпорност на кварове. Ако једно складиште података доживи квар или прекид, услуге се могу преусмерити на друга складишта података, чиме се одржава доступност услуга. Такође, дистрибураност обезбеђује опоравак од катастрофе и заштиту података у случају хаварије или губитка података у једном складишту података прављењем резервних копија на различитим локацијама.

Битна карактеристика архитектуре облака је и њена скалабилност. Неопходно је да се ресурси могу лако повећати или смањити како би се прилагодили променљивим потребама корисника, што се постиже кроз концепт скалабилности. Хоризонтална скалабилност подразумева додавање или уклањање додатних виртуелних машина или других ресурса како би се променио капацитет обраде захтева и отпорност на оптерећење. Вертикална скалабилност подразумева повећање ресурса, као што су процесорска снага и меморија, унутар постојећих виртуелних машина како би се побољшале перформансе апликација. Како би корисничко искуство било што боље битно је да сваки кориснички захтев који укључује скалирање ресурса буде брз и поуздан.

Још један важна област архитектуре облака је њена безбедности. Безбедност у облаку представља поддомен области као што су безбедност мреже, рачунара, безбедност информација. Обухвата различите полисе, регулативе, стандарде и технологије које контролишу пренос података и апликација као и њима додељене рачунарске инфраструктуре рачунарског облака. Она подразумева имплементацију сигурносних механизама попут енкрипције података, аутентификације и ауторизације, заштите мреже и контроле приступа како би се подаци заштитили од неовлашћеног приступа или напада. Пружалац услуга облака треба да обезбеди маскираност податка како би само ауторизовани корисници могли имати приступ подацима у читљивом облику. Дигитални идентитети и креденцијали морају бити заштићени као и било који податак који пружалац услуга сакупља или настаје корисниковом активношћу у облаку. Приступ подацима корисника мора бити документован, на нивоу сваке машине на којој су ускладиштени подаци тог корисника. Поред праћења логова и ревизије путање пружаоци услуга у договору са корисницима обезбеђују потребну сигурност података и чувају их онолико колико корисник то захтева.

3.6 Уговор о нивоу услуге

Уговор о нивоу услуге (енгл. *Service Level Agreement, SLA*) је правни документ који дефинише обавезе и очекивања између пружаоца услуга у облаку и корисника у погледу квалитета, доступности и перформанси услуге [18]. Дефинишу се параметри, као што су доступност, време одзива, брзина преноса података и сигурност, на којима се затим мери квалитет услуге. Овај уговор је важан инструмент који помаже да обе стране разумеју очекивања, обезбеде квалитетну услугу и има за циљ обезбеђивање транспарентности, поузданости и одговорности у погледу пружања услуга у окружењу облака као и механизме за решавање спорова у случају неиспуњавања уговорних обавеза [19].

Овај уговор најчешће садржи следеће елементе:

- дефиниције – дефинишу се кључни термини који се користе у уговору како би се обе стране сложиле о њиховом значењу;
- обухват услуге – прецизан опис услуге коју пружа пружалац услуге у облаку, укључујући све функционалности, могућности и ограничења;
- нивои услуге – дефинишу се нивои квалитета који се очекује од услуге, као што су минимална доступност, време одзива и перформансе система који могу бити изражени у процентима, временским интервалима или другим погодним мерама;
- одговорности и обавезе – прецизирају се одговорности и обавезе за обе стране, укључујући обвезу пружаоца услуге да одржава и обезбеди услугу, као и обавезе корисника у погледу плаћања, безбедности података и начина коришћења услуге;
- надокнаде за кршење уговора – дефинишу се надокнаде које ће пружалац услуге платити кориснику у случају кршења уговорних обавеза у вези са нивоом услуге;
- поступци праћења и извештавања – дефинишу детаље о начину праћења нивоа услуге и извештавања о њиховом испуњењу, укључујући редовне извештаје или механизме за решавање спорова.

Кључни елементи уговора о нивоу услуге подразумевају опсег и доступност услуге, време одзива, нивое подршке и време решавања проблема, пенале и надокнаде у случају неиспуњења услова, као и процедуре за решавање спорова. Метрикама се обично дефинишу доступност услуге (на пример, услуга ће бити доступна 99,9% времена), време одзива (на пример, максимално време потребно за добављача услуге да одговори на упите или захтеве корисника), брзина преноса података, време решавања проблема, доступност техничке подршке и слично. Такође, дефинишу се и пенали или надокнаде које се примењују у случају непоштовања уговорених нивоа услуге, као што су новчане казне, додатне услуге и повлашћења или други облици компензације за обе стране.

3.7 Одржавање облак окружења

Облак окружење (енгл. *cloud environment*) је виртуелно окружење у којем се корисницима пружају различите услуге и могућности преко интернета и односи се на инфраструктуру, ресурсе и услуге које су доступне у рачунарству у облаку.

Редовно одржавање облак окружења је кључно за обезбеђивање сигурности и стабилности података, као и оптималних перформанси облака. Ово подразумева редовну примену сигурносних закрпа и исправки, као и надоградњу софтверских и хардверских компонената и оперативних система. Такође, како би се идентификовали потенцијални проблеми или уска грла неопходно је континуирано надгледање перформанси инфраструктуре које подразумева праћење ресурса, мрежних веза, оптерећења система и доступности услуга.

Одржавање облак окружења захтева добру координацију између пружаоца услуга и корисника уз праћење услова дефинисаних уговором о нивоу услуге и осигурањем минималног утицаја на континуитет пословања и перформансе услуга [17]. Изазов одржавања је координација ажурирања, сигурносних закрпа, креирања резервних копија и других операција које могу узроковати прекиде или које утичу на перформансе и минимизација утицаја ових операција на доступост услуге крајњим корисницима. У облак окружењу, пружаоци услуга стално раде на креирању што оптималнијег плана одржавања који има за циљ повећање доступности и поузданости облак окружења уз ефикасно управљање ажурирањем софтвера и инфраструктуре.

Начини оптимизације одржавања подразумевају одређене акције које омогућавају да из угла корисника нема прекида током одржавања или да је трајање прекида минимално како би се што боље испоштовали услови доступности дефинисани у уговору о нивоу услуге. Пружаоци услуга облак окружења континуирано раде на унапређивању ове области у циљу обезбеђивања високог нивоа доступности услуга и минималног утицаја током извршавања датих операција.

Скупови доступности

Скупови доступности (енгл. *Availability Sets*) су механизам који се користи у неким облак окружењима за обезбеђивање високе доступности и поузданости апликација и услуга које се извршавају у облаку. Они подразумевају распоређивање виртуелних машина на различите хост рачунаре како би се обезбедила редундантност, минимизовало време прекида и повећала доступност.

Скуп доступности подразумева груписање виртуелних машина које пружају исту услугу или су међусобно повезане. Виртуелне машине у оквиру истог скупа доступности се затим распоређују на различите хост рачунаре што спречава да се сви ресурси апликације налазе на једном месту, чиме се смањује ризик од прекида услуге због квара на хардверу или других проблема[2]. Ако дође до квара на неком од хост рачунара, виртуелне машине се аутоматски пребацују на друге доступне хост рачунаре, минимизујући прекид услуге. Такође, када је потребно ажурирати хост рачунар, виртуелне машине ће бити померене на друге доступне хост рачунаре пре почетка одржавања, чиме ће дужина прекида бити минимизована и износиће пар секунди уместо врмена потребног за ажурирање хост рачунара.

Коришћење скупова доступности је пракса препоручена за све апликације и услуге које захтевају високу доступност и минимално време прекида услуге. Пружаоци услуга облака обично обезбеђују алате и функционалности за управљање скуповима доступности и аутоматизацију пребацивања виртуелних машина у случају квара или планираног одржавања.

Период одржавања

Минимизација времена прекида коју обезбеђују скупови доступности једна је од оптимизација одржавања облак окружења. Без обзира на унапређења која ова оптимизација нуди, често чак и минимални прекиди могу имати велики утицај на пословање корисника, због чега је битно да се поред трајања прекида оптимизује и тренутак у коме ће се тај прекид десити што се дефинише периодом одржавања.

Период одржавања у контексту облак окружења представља временски интервал током којег се изводе планиране активности одржавања, било система или инфраструктуре, како би се обезбедио исправан рад и оптималне перформансе облак окружења. Обично се период одржавања заказује унапред, уз поштовање услова дефинисаних у уговору о нивоу услуге. Како није лако планирати тачно време потребно за различите типове одржавања, као ни тачан тренутак када ће бити неопходно извршити одржавање најчешће се у уговору о нивоу услуге договара само број планираних одржавања у одређеном временском периоду и ниво доступности услуге, а не и сам тренутак када ће се одржавања десити. Због тога, период одржавања треба пажљиво планирати и координисати са корисницима како би се његов утицај на кориснике минимизовао.

Неки од пружаоца услуга облак окружења нуде корисницима и могућност избора између пар различитих модела предефинисаних периода одржавања. На пример, *Microsoft Azure* подразумевано блокира планирана одржавања у периоду од 08:00 до 17:00 по локалном времену сваког дана како би избегли прекиде током типичних радних сати корисника. Поред овог подразумеваног ограничења, у понуди су још два предефинисана периода одржавања које корисници могу одабрати, планирано одржавање радним данима, где се одржавање може вршити од 22:00 до 06:00 по локалном времену од понедељка до четвртка, односно планирано одржавање викендом, где се одржавање може вршити од 22:00 до 06:00 по локалном времену од петка до недеље. Када се изврши избор периода за одржавање и заврши конфигурација услуге, планирана одржавања ће се одвијати само током изабраног периода [4].

Током периода одржавања најчешће се спроводе ажурирања софтвера и инфраструктуре, као и замена или надоградња хардвера. Могу се вршити и друге врсте планираних одржавања као што су аутоматска проширења или смањења капацитета уколико корисници изаберу такву опцију, као и

прављења резервних копија и друге операције које не узрокују прекиде, али могу утицати на перформансе облак окружења.

Избор најбољег времена за одржавање

Претходно наведени приступи пружају већу флексибилност и бољу доступност сервиса корисницима, али не користе расположиве информације о претходном раду корисника које би могле помоћи при избору најбољег времена за одржавање.

Како би се одабрало најбоље време за одржавање треба изабрати време када је најмање корисника активно на систему [14]. Међутим, у облак окружењу најчешће корисници облак окружења, односно компаније или организације које користе рачунарство у облаку, имају своје кориснике чију активност пружаоци услуга облак окружења не могу лако директно пратити. Тада анализа статистике коришћења услуга и степен искоришћености ресурса може помоћи у одређивању најбољег времена за одржавање.

Неке од ствари на које треба посебно обратити пажњу су периоди у којима се достижу максимуми коришћења и најмање активни периоди како би се идентификовали временски интервали када је најмање вероватно да ће прекид услуге имати значајан утицај на кориснике облак окружења и њихове кориснике. Такође, треба се водити рачуна о географској распрострањености и временским зонама, односно радним сатима важних клијената, као и локалних празника, специјалних датума или периода када корисници могу имати већу активност, чиме би прекид имао већи утицај на њихово пословање.

У наставку овог рада биће описана једна идеја решавања овог проблема која се базира на посматрању процента искоришћености процесора током времена, налажењу периода најмање активности и избору пронађеног периода најмање активности као будућег периода одржавања.

Глава 4

Проналажење периода најмање активности процесора

Као што је поменуто у глави 3 проналажење периода најмање активности и одабир тог периода као најбољег периода за одржавање може допринети томе да одржавње има мањи утицај на кориснике, због чега је очекивано да задовољство корисника пруженом услугом у том случају буде веће. У наставку ће фокус бити на решавању проблема избора најбољег времена за одржавање јавног облака чији је модел пружања услуга платформа као сервис.

Облак окружење састоји се од хост рачунара на којима се налази много виртуелних машина, где свака од њих има различито радно оптерећење, искоришћеност процесора, искоришћеност меморије, перформансе и слично. Виртуелне машине током времена могу бити динамички креиране и брисане на хост рачунару, а може доћи и до њиховог померања између више хост рачунара. У случају јавног облака сваку од виртелних машина треба третирати као црну кутију, с обзиром да пружаоци услуга облак окружења немају увид у тачне операције које корисници извршавају на својим ресурсима. Због тога најчешће не постоји информација за које виртуелне машине је неопходно одржати најбоље перформансе у ком тренутку, због чега се сматра да су за одређеног корисника све виртуелне машине које поседује једнако битне, осим у случају да постоји информација која нам говори супротно. Као мера коришћења ресурса биће коришћена искоришћеност процесора која ће бити посматрана у циљу описивања и уочавања образаца понашања, а као најбоље време за одржавање биће изабран период када је активност процесора најмања.

4.1 Дефиниција проблема

Проблем избора најбољег времена за одржавање има два дела који су веома слични, али могу имати различите приступе решавању проблема. Један део се односи на одржавање хост рачунара, односно физичких сервера, док се други односи на одржавање виртуелних машина.

Део проблема који се односи на избор времена за одржавање виртуелних машина састоји се од анализе временске серије која се односи на искоришћеност процесора конкретне виртуелне машине, а затим на избору најбољих n сати, где је n број који треба бити конфигурабилан. Различите операције могу захтевати различиту дужину времена одржавања, због чега треба минимизовати утицај на корисника што је боље могуће избором само онолико сати колико је потребно за дату операцију одржавања.

Проблем избора најбољег времена ажурирања хост рачунара је нешто компликованији. У овом случају не може се лако урадити агрегирање података за све виртуелне машине које се налазе на истом хост рачунару јер је дефиниција услова за избор најбољег времена за одржавање хост рачунара комплексан проблем због динамичких померања виртуелних машина са једног хост рачунара на други. Идеално би се исте виртуелне машине увек заједно померале са једног на други хост рачунар и остајале груписане, или би виртуелне машине које се налазе на истом хост рачунару имале сличне узорке понашања, али нажалост најчешће ниједна од поменутих ствари није тачна. Један приступ овом проблему могао би бити да се посматра искоришћеност процесора самог хост рачунара, када се проблем своди се на варијанту проблема са виртуелним машинама. Други приступ могао би бити да се пронађе неки временски период када би у просеку утицај на кориснике чије се виртуелне машине у том тренутку налазе на хост рачунару био најмањи, када би се посматрао степен искоришћености процесора сваке виртуелне машине на хост рачунару. У овом случају би могло да се говори и о тежинама за сваку од виртуелних машина, на пример у случају да постоји неки веома битан корисник за кога је битније минимизовати утицај одржавања, чак и по цену да неки други корисник буде мање задовољан изабраним временом. Могуће су и разне друге модификације које би могле да се искористе при дефинисању начина на који се бира најбољи временски период за одржавање. У наставку ће бити речи о решењима базираним на претходно описаним двема идејама.

4.2 Приказ коришћених Python библиотека

Ово поглавље биће кратак пролаз кроз важне библиотеке и функције које ће бити коришћене приликом имплементације, како би објашњења у наставку била лакша за праћење. Како *matplotlib* [3] и *NumPy* [5] немају функције које је битно појаснити пре саме имплементације њихова објашњења ће бити изостављена у овом делу.

Pandas

Библиотека *pandas* пружа подршку за коришћење структура података које имају за циљ да буду основни градивни блок за практичну анализу података и рад са њима. Ова библиотека нуди две примарне структуре података *Series* и *DataFrame*, од којих ће у наставку бити коришћена структура *DataFrame*. Структура података *DataFrame* садржи означене осе (редове и колоне) и омогућава лак рад са дводимензионим подацима [6]. Такође, ова библиотека пружа подршку и за читање CSV фајлова коришћењем функције *pandas.read_csv* која ће бити коришћена за читање и парсирање скупа података.

SciPy

SciPy представља колекцију математичких алгоритама и практичних функција која кориснику пружа команде и класе високог нивоа за манипулацију и визуелизацију података [8]. У наставку ће од значаја бити *scipy.ndimage* и *scipy.signal*.

Својства *scipy.ndimage* која ће бити битна у наставку јесу функције за линеарно и нелинеарно филтрирање и изравнање које су дизајниране да раде са низовима произвољне димензионалности. Функција *scipy.ndimage.gaussian_filter* имплементира вишедимензионални Гаусов филтер који врши изравнање временске серије.

За обраду сигнала користи се *scipy.signal*. У наставку ће бити коришћена функција *scipy.signal.find_peaks* која као параметар захтева једнодимензионални низ, а као резултат проналази све локалне максимуме поређењем суседних вредности. Резултујући врхови не морају бити сви локални максимуми,

већ је могуће задати услове које максимум треба да испуни да би се сматрао врхом и на тај начин добити тражени подскуп локалних максимума.

Statsmodel

Модул *statsmodel* обезбеђује класе и функције за креирање многих статистичких модела, тестова и истраживање података. Овај модул подржава спецификацију модела користећи моделе налик онима у R програмском језику и рад са pandas *DataFrame* структурама [9].

У наставку ће *statsmodels.tsa.holtwinters.ExponentialSmoothing* бити коришћен за Холт-Винтерсово[1] експоненцијално сезонско изравнавање и проналажење оних временских серија које имају константне вредности, односно чије вредности варирају мање од одређене унапред задате вредности.

Такође, *statsmodels.tsa.statespace.sarimax.SARIMAX* биће коришћен за креирање и ретренирање сезонског ARIMA модела. Овај модел подржава учитавање сачуваних параметара као почетних параметара приликом креирања модела који ће затим бити даље прилагођавани.

Pmdarima

Библиотека *pmdarima* је *Python* библиотека чији је фокус анализа временских серија і ARIMA модели. Еквивалентна је подршци коју пружа *auto.arima* у програмском језику R [7].

Коришћењем *pmdarima.arima.auto_arima* биће креиран ARIMA модел, његови параметри биће оцењени, а затим и сачувани како би могли касније бити коришћени приликом прављења претходно поменутог SARIMAX модела. Могуће је користити обичну и сезонску варијанту ARIMA модела, као и задати почетне и максималне вредности параметара чиме се може обезбедити боље прилагођавање модела подацима уколико је то потребно.

Из ове библиотеке ће за рачунање реда диференцирања d, односно D сезонског ARIMA модела бити коришћене функције *pmdarima.arima.ndiffs* и *pmdarima.arima.nsdiffs* које тестирањем стационарности за различите вредности d, odnosno D рачунају вредност параметра за коју је временска серија стационарна.

[1] Holt-Winters

4.3 Скуп података

Скуп података за овај проблем могао би се састојати од информација о искоришћености процесора током времена за сваку виртуелну машину или за сваки хост рачунар. Искоришћеност процесора једне виртуелне машине представља једну временску серију са непрекидним временом. Мерења се не могу вршити непрекидно, али могуће је добити информације о искоришћености процесора у практично сваком временском тренутку јер се мерења врше толико често да основни скуп података садржи временске серије са временом које је практично непрекидно. Како је број мерења за једну виртуелну машину огроман, мери се у десетинама хиљада података дневно, а онда све тај број треба још помножити са бројем виртуелних машина чији се број такође може мерити у хиљадама долазимо до броја података које је веома тешко обрадити у разумном времену. С обзиром на количину података над којима се ради и време потребно за тренирање, ретренирање и предвиђање модела који користи све податке које можемо узети у обзир, добијени резултати би били практично неупотребљиви у тренутку када буду израчунати, односно скоро увек би се односили на прошле временске тренутке.

Како би поменути скуп био прилагођен тако да буде довољно детаљан за потребе израчунавања најбољег времена за одржавање, али и такве величине да је могућа његова обрада и добијање резултата у разумном времену извршено је временско агрегирање и добијена дискретна временска серија која садржи податке о искоришћености процесора агрегиране на сат времена на различите начине.

Најпре је потребно дефинисати функције за учитавање и парсирање скупа података из CSV фајла и извршити учитавање података.

Код 4.3.1

```
def load_data(filename):
    data=pandas.read_csv(filename,
                         parse_dates=True,
                         index_col=1,
                         squeeze=True,
                         date_parser=parser)
    return data
```

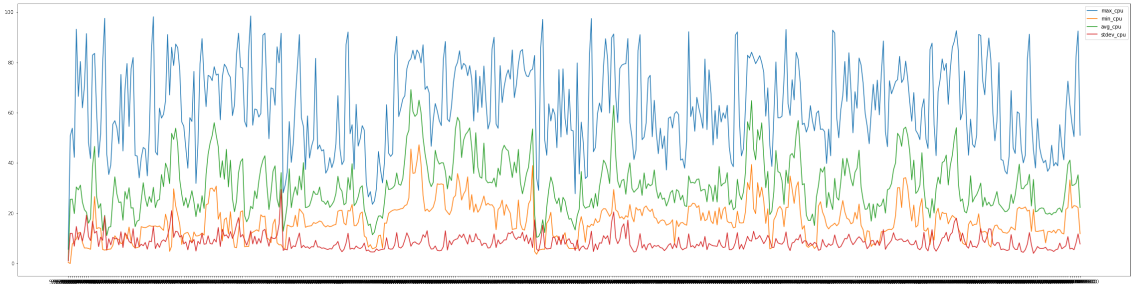
```
def parser(x):
    return datetime.datetime.strptime(str(x),'%m/%d/%y %H:%M')
```

На слици 4.1 приказан је део скупа података над којим ће бити рађено у наставку. Скуп података садржи име виртуелне машине ***VMName***, име хоста ***HostName***, време ***Hour*** у коме је очитана вредност агрегирано тако да се односи на почетак сата како би избор времена за одржавање био такав да одржавање увек почиње на пун сат и траје n сати, где је n конфигурабилно. Затим овај скуп садржи и агрегиране податке о искоришћености процесора на више начина, односно ***max_ cpu*** који означава максимални проценат искоришћености процесора током посматраног сата, ***min_ cpu*** који означава минимални проценат искоришћености процесора током посматраног сата, ***avg_ cpu*** који представља просечну вредност искоришћености процесора током посматраног сата и ***stdev_ cpu*** који представља стандардну девијацију.

	VMName	Hour	HostName	max_cpu	min_cpu	avg_cpu	stdev_cpu
0	cdf2b10c1ed8	9/9/2019 14:00	071E9CEE54E2A2388C1A6980BED9F68F7D7304BC73DFCE...	5.636008	0.438370	1.066536	0.981527
1	cdf2b10c1ed8	9/9/2019 15:00	071E9CEE54E2A2388C1A6980BED9F68F7D7304BC73DFCE...	50.704308	0.006659	25.494145	12.002797
2	cdf2b10c1ed8	9/9/2019 16:00	071E9CEE54E2A2388C1A6980BED9F68F7D7304BC73DFCE...	53.820972	7.722448	25.630070	11.958147
3	cdf2b10c1ed8	9/9/2019 17:00	071E9CEE54E2A2388C1A6980BED9F68F7D7304BC73DFCE...	42.237447	8.748336	19.862113	6.896077
4	cdf2b10c1ed8	9/9/2019 18:00	071E9CEE54E2A2388C1A6980BED9F68F7D7304BC73DFCE...	93.201360	9.855354	30.452347	14.734517

Слика 4.1: Скуп података.

Посматрани скуп података садржи информације о искоришћености процесора у претходних 90 дана. На слици 4.2 приказани су агрегирани подаци за једну временску серију за период од 30 дана. Плавом бојом приказан је *max_ cpu*, наранџастом *min_ cpu*, зеленом *avg_ cpu* и црвеном *stdev_ cpu*. Како је идеја овог рада предвиђање најбољег времена за одржавање, биће посматрано колика је употреба процесора у просечном случају, односно у наставку фокус ће бити на посматрању и описивању *avg_ cpu*.



Слика 4.2: Приказ различитих варијаната агрегације временске серије.

4.4 Имплементација

Агрегирањем скупа података уштеђено је одређено време, међутим током имплементације примећено је да та уштеда није довољна. Тренирање, односно ретренирање ARIMA модела трају предуго када се за све временске серије користи овај модел. Као што је већ поменуто, број временских серија над којима је очекивано да модел ради мери се хиљадама, због чега је идеја да се пре коришћења самог модела најпре покуша детектовање најбољих периода за одржавање другим, „јефтинијим" методама у смислу времена.

Произвољни избор времена одржавања

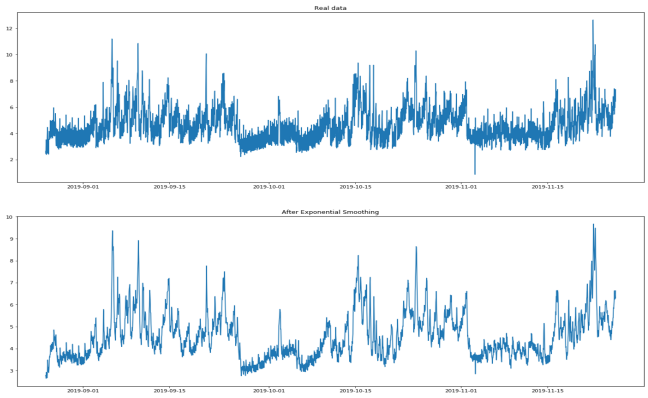
Прва идеја је да се издвоје оне временске серије које имају вредности које су константне током времена, односно оне чија вредност варира мање од унапред дефинисане вредности, као и оне код којих је максимум искоришћености процесора у претходном периоду мањи од унапред дефинисане минималне вредности. Постојање овакве временске серије нам говори да је употреба процесора конкретне виртуелне машине често константна и да избор и предвиђање одређеног периода одржавања не би донели битна побољшања у искуству корисника.

Такође, постоје и случајеви новокреираних виртуелних машина за које нема смисла вршити предвиђања због недостатка података на основу којих би било могуће предвидети њено будуће понашање. Може се сматрати да нема смисла вршити предвиђања за виртуелне машине за које не постоје подаци за најмање претходних 7 дана, односно 168 сати.

За временске серије које испуне претходне услове сматрамо да је одржавање могуће у било ком тренутку. Подразумеване вредности функције *can_be_updated_anytime* која проверава претходна својства су да временску серију можемо сматрати недовољно активном ако је максимум њене искоришћености процесора мањи од 5%, односно можемо је сматрати константном ако је максимална промена у активности процесора током читавог периода за који посматрамо претходне вредности мања од 3% након примене експоненцијалног изравнања.

Код 4.4.1

```
def can_be_updated_anytime(data, min_length=168, lowest_max=5,
                                        smoothed_range=3):
    if len(data)<min_length:
        return True
    if data.avg_cpu.max()<lowest_max:
        return True
    model=statsmodels.tsa.holtwinters.ExponentialSmoothing(
                        data.avg_cpu).fit()
    predictions=model.predict(start=data.index[0],
                                    end=data.index[-1])
    if numpy.max(predictions)<smoothed_range or
     numpy.max(predictions)-numpy.min(predictions)<smoothed_range:
        return True
    else:
        return False
```



Слика 4.3: Пример временске серије пре и након експоненцијалног изравнања.

На слици 4.3 приказан је један пример временске серије из скупа података. На првом делу графички су приказане почетне вредности посматране временске серије, док су на другом делу приказане впредности исте временске серије након примене експоненцијалног изравнања.

Дневни периоди

Уколико се у посматраној временској серији обрасци понашања периодично понављају сваких 24 сата, тада сматрамо да временска серија има дневне периоде. У овом случају је избор временског тренутка у току једног дана када се оджавање треба десити значајан, док се дан у недељи може изабрати произвољно, односно по потреби можемо користити исти изабрани период за било који дан у истој недељи.

Код 4.4.2

```
def classify_peaks_of_array(array,n=24,error=1):
    array_peaks=scipy.signal.find_peaks(array)[0]
    peaks=numpy.in1d(numpy.mod(array_peaks,n),numpy.arange(-error,
                                        error+1,1))
    return peaks

def auto_correlation(array, lag=1):
    return numpy.corrcoef(array[lag:],array[:array.shape[0]-lag])[0,1]

def check_daily_period(data,data_length=168,number_of_peaks=6):
    df=data[(data.index>data.index.max()-datetime.timedelta(hours=
                                        data_length))]
    if len(df)<data_length:
        return False
    x=scipy.ndimage.gaussian_filter(df.avg_cpu, sigma=2)
    lag = numpy.arange(0, data_length-1, 1)
    acr = [auto_correlation(x, lag=l) for l in lag]
    daily_peaks=classify_peaks_of_array(acr,24).astype(np.int).sum()
    if daily_peaks==number_of_peaks:
        return True
    else:
        return False
```

Приликом тражења дневних периода најчешће је довољно посматрати податке за последњих недељу дана, односно 168 сати, како би било могуће упоредити дневну активност процесора са вредностима које је имала током сваког од дана претходне недеље. За проналазак дневних периода користи се функција *check_ daily_ period* која за израчунавање мере сличности временске серије са њеним претходним вредностима користи аутокорелациону функцију *auto_ correlation*. Најпре се елиминише шум користећи Гаусово филтрирање, затим примени поменута аутокорелациона функција над филтрираним вредностима, након чега се над добијеним резултатом користећи *classify_ peaks_ of_ array* траже они врхови који представљају локалне максимуме међусобно удаљене 24 сата. На дужини података од 168 сати тражи се 6 таквих врхова како би се сматрало да временска серија има дневне периоде. Графички приказ сваког од поменутих корака, као и добијени резултати приказани су у поглављу 4.5.

Недељни периоди

Слично као за дневне периоде, уколико се у временској серији обрасци понашања понављају сваких 7 дана, односно 168 сати, тада сматрамо да временска серија има недељне периоде. Недељни периоди говоре да је без обзира на посматрану недељу одговарајуће време одржавања дефинисано избором дана у датој недељи и временског тренутка у току изабраног дана.

Имплементација је веома слична као у случају дневних периода, са разликом да се овде посматрају подаци за претходних 6 недеља, односно 1008 сати, и траже узорци који се понављају сваких 7 дана, односно 168 сати. Ово је неопходно да би било довољно података који би омогућили уочавање периодичности. За проналазак недељних периода користи се функција *check_ weekly_ period*. Једина разлика у имплементацији у односу на претходно описану функцију *check_ daily_ period* је то што се за проналажење врхова користи *classify_ peaks_ of_ array* на начин да сматрамо да имамо недељне периоде уколико се врхови који представљају локалне максимуме налазе на удаљености од 168 сати и, с обзиром да се посматрају подаци за последњих 6 недеља, очекивани број таквих врхова је 5. Као и у случају дневних периода, графички приказ претходно описаних корака и добијени резултати приказани су у поглављу 4.5.

Код 4.4.3

```
def classify_peaks_of_array(array,n=168,error=1):
    array_peaks=scipy.signal.find_peaks(array)[0]
    peaks=numpy.in1d(numpy.mod(array_peaks,n),numpy.arange(-error,
                                            error+1,1))
    return peaks

def auto_correlation(array, lag=1):
    return numpy.corrcoef(array[lag:],array[:array.shape[0]-lag])[0,1]

def check_weekly_period(data,data_length=1008,number_of_peaks=5):
    df=data[(data.index>data.index.max()-datetime.timedelta(hours=
                                            data_length))]
    if len(df)<data_length:
        return False
    x=scipy.ndimage.gaussian_filter(df.avg_cpu, sigma=2)
    lag = numpy.arange(0, data_length-1, 1)
    acr = [auto_correlation(x, lag=l) for l in lag]
    weekly_peaks=classify_peaks_of_array(acr,168).astype(np.int).sum()
    if weekly_peaks==number_of_peaks:
        return True
    else:
        return False
```

Сезонски ARIMA модел

У случају да ниједна од претходно уведених једноставнијих операција није довела до резултата, односно уколико временска серија није нова, константна, неактивна и не садржи дневне и недељне периоде предвиђања за ту временску серију биће вршена коришћењем сезонског ARIMA модела, $ARIMA(p,d,q) \times (P,D,Q)_s$. Како би било могуће користити сезонски ARIMA модел најпре је потребно оценити његове параметре p (ред ауторегресионе компоненте), q (ред компоненте покретних просека), d (ред диференцирања), P (ред сезонске ауторегресионе компоненте), Q (ред сезонске компоненте покретних просека), D (ред сезонског диференцирања) и s (период сезоне).

Код 4.4.4

```
def arima_coefs(data,m=24,alpha=0.01,maxiter=100):
    d=pmdarima.arima.ndiffs(data,alpha=0.01)
    D=pmdarima.arima.nsdiff(data,m=m)
    model=pmdarima.arima.auto_arima(data, suppress_warnings=True, m=m,
                                    trend='ct', alpha=alpha,
                                   maxiter=maxiter, d=d, D=D)
    order=model.order
    seasonal_order=model.seasonal_order
    return (order, seasonal_order)
```

За оцењивање параметара сезонског ARIMA модела биће коришћена функција *auto_arima*. Приликом тестирања, рачунањем средње квадратне и средње апсолутне грешке, утврђено је да функција *auto_arima* даје боље резултате приликом рада са унапред задатим вредностима параметара d и D. Због тога ће вредности параметара d и D најпре бити оцењени коришћењем функција *ndiffs* и *nsdiffs*, а затим прослеђени функцији *auto_arima* која ће вршити оцењивање осталих параметара модела. Да бисмо обезбедили да параметар тренда може бити и константан и линеаран, као вредност параметра *trend* функције *auto_arima* биће прослеђена вредност *'ct'*.

Како је тренирање, односно оцењивање параметара модела из угла времена веома скупа операција ова операција биће извршена само на почетку, односно у случајевима када креирани модел више није релевантан, док ће у осталим случајевима бити рађено само ретренирање, односно прилагођавање већ оцењених параметара модела новодобијеним подацима. Најчешће је потребно резултате добити брзо због чега је идеја да се израчунати параметри чувају како би могли бити коришћени онда када су потребни за брзо креирање модела.

За креирање модела у случају већ израчунатих параметара биће коришћена функција *make_model* која креира сезонални ARIMA модел користећи библиотечку подршку коју пружа SARIMAX који има добру подршку за креирање модела на основу доступних, већ оцењених, параметара. Овај модел се затим прилагођава новим подацима након чега ће бити коришћен за предвиђање будућих вредности.

Код 4.4.5

```
def make_model(data, params,enforce_stationarity=True,
                                        enforce_invertibility=True):
    model = statmodels.tsa.statespace.sarimax.SARIMAX(data,
                    order=params[0],
                    seasonal_order=params[1],
                    trend='ct',
                    simple_differencing=True,
                    enforce_stationarity=enforce_stationarity,
                    enforce_invertibility=enforce_invertibility)
    model_fit=model.fit()
    return model_fit
```

Функција *make_predictions* као резултат враћа предвиђања за наредних n+h сати. Додавање h, које означава број сати потребних за одржавање, је неопходно како би се обезбедило да као почетак одабраног временског периода за одржавање може да се јави било који од n временских тренутака за које вршимо предвиђања.

Код 4.4.6

```
def make_predictions(model, n, h):
    return model.forecast(n+h)
```

Одабир најбољег времена за одржавање

Након добијених предвиђања будућих вредности серије треба одредити које би време било најбоље за одржавање. Као што је поменуто у поглављу 3.7 најчешћа пракса је да се одржавања не врше током радних сати дефинисаних као период од 08:00 do 17:00 по локалном времену [4]. Такође, још једна пракса је да су сви записи времена представљени у UTC временској зони. Ово значи да је за конвертовање у локално време неопходно знати колико је сати оно испред или иза UTC временске зоне.

Функција *find_n_h* као параметре има предвиђања добијена за посматрану временску серију, број сати потребних за одржавање, датум последње

вредности у скупу података, као и број који означава временску разлику између UTC временске зоне и локалног времена, а као резултат враћа почетак и крај пронађеног најбољег времена за одржавање. У случају да време не треба рачунати на основу локалног времена, већ UTC временске зоне или су подаци већ пребачени у локално време, последњи параметар се може изоставити.

Код 4.4.7

```
def find_n_h(predictions,n,date,timezone=0):
    start=date+datetime.timedelta(hours=1+timezone)
    end=date+datetime.timedelta(hours=n+1+timezone)
    j=0
    while (start.time()>datetime.time(8,0,0)
           and start.time()<datetime.time(17,0,0))
           or (end.time()>datetime.time(8,0,0)
           and end.time()<datetime.time(17,0,0)):
        start=start+datetime.timedelta(hours=1)
        end=end+datetime.timedelta(hours=1)
        j=j+1
    sum_n_h=sum(predictions[j:j+n])
    minimum=sum_n_h
    min_start=start
    min_end=end
    for i in range(j+n,len(predictions)):
        sum_n_h=sum_n_h+predictions[i]-predictions[i-n]
        if sum_n_h<minimum:
            start=date+datetime.timedelta(hours=i-n+2+timezone)
            end=date+datetime.timedelta(hours=i+2+timezone)
            if (start.time()>datetime.time(8,0,0)
                   and start.time()<datetime.time(17,0,0))
                   or (end.time()>datetime.time(8,0,0)
                   and end.time()<datetime.time(17,0,0)):
                continue
            minimum=sum_n_h
            min_start=start
            min_end=end
    min_start=min_start+datetime.timedelta(hours=-timezone)
    min_end=min_end+datetime.timedelta(hours=-timezone)
    return [min_start,min_end]
```

Проналазак најбољег времена за одржавање имплементиран у функцији *find_n_h* представља проналазак периода од n сати који има просечно најмању искоришћеност процесора. Проналазак овог периода врши се рачунањем средње вредности искоришћености процесора за сваки од периода дужине n сати који се налази у предвиђањима која представљају први параметар поменуте функције. Тренутна имплементација поштује радне сате по локалном времену, због чега се рачунање врши само за оне периоде од n сати који се у потпуности налазе ван периода од 08:00 до 17:00 по локалном времену.

Уколико не би било потребе поштовати радне сате по локалном времену све провере које се односе на то да ли је добијени период одржавања у периоду између 17:00 и 08:00 по локалном времену могле би бити прескочене. Такав приступ би могао бити користан за оне кориснике који користе центре података које се налазе у другој временској зони у односу на временску зону у којој корисници послују. Најчешће такви корисници имају знатно другачије обрасце активности процесора током времена у односу на оне кориснике који се налазе у истој временској зони као и центар података. Неопходно је само напоменути да је у сваком тренутку неопходно поштовати све већ уговорене услове са корисником, те би у случају непоштовања радних сати било потребно дефинисати нови модел периода одржавања који би корисник могао да изабере за будућа планирана одржавања.

Све о чему је до сада било речи се односило на одржавање и хост рачунара и виртуелних машина. Као што је поменуто на почетку та два проблема јесу веома слична, поготово у ситуацији да хост рачунар посматрамо као једну машину са дефинисаном мером активности процесора током времена, када би било могуће применити функцију *find_n_h* над подацима који представљају искоришћеност процесора тог хост рачунара током времена. Други случај био би када не постоје информације о активности процесора хост рачунара, већ израчунавања треба вршити на основу активности процесора виртуелних машина које се налазе на посматраном хост рачунару.

Једна идеја решења проблема одржавања хост рачунара када немамо информација о активности његовог процесора могла би бити да се на основу активности процесора виртуелних машина рекреира активност процесора хост рачунара. Проблем који се може јавити у том случају јесте мера удела процесора хост рачунара који свака виртуелна машина користи, као и других конфигурација како хост рачунара, тако и виртуелних машина, због чега ово ре-

шење није добро дефинисано и оптимално за разматрање у овом раду. Друга идеја могла би бити да се активност процесора свих виртуелних машина које се налазе на хост рачунару посматра одвојено и над тим подацима пронађе неки просечно најбољи период што је дефинисано у функици *find_n_h_2*.

Код 4.4.8

```
def find_n_h_2(prediction,n,date,timezone=0):
    start=date+datetime.timedelta(hours=1+timezone)
    end=date+datetime.timedelta(hours=n+1+timezone)
    j=0
    while(start.time()>datetime.time(8,0,0)
          and start.time()<datetime.time(17,0,0))
          or (end.time()>datetime.time(8,0,0)
          and end.time()<datetime.time(17,0,0)):
        start=start+datetime.timedelta(hours=1)
        end=end+datetime.timedelta(hours=1)
        j=j+1
    sum_n_h=0
    for p in prediction:
        sum_n_h=sum_n_h+sum(p[j:j+n])
    minimum=sum_n_h
    min_start=start
    min_end=end
    for i in range(j+n,len(prediction[0])):
        for p in prediction:
            sum_n_h=sum_n_h+p[i]-p[i-n]
        if sum_n_h<minimum:
            start=date+datetime.timedelta(hours=i-n+2+timezone)
            end=date+datetime.timedelta(hours=i+2+timezone)
            if (start.time()>=datetime.time(8,0,0)
                  and start.time()<datetime.time(17,0,0))
                  or (end.time()>datetime.time(8,0,0)
                  and end.time()<=datetime.time(17,0,0)):
                continue
            minimum=sum_n_h
            min_start=start
            min_end=end
    min_start=min_start+datetime.timedelta(hours=-timezone)
    min_end=min_end+datetime.timedelta(hours=-timezone)
    return [min_start,min_end]
```

До сада је поменуто неколико идеја категоризације временских серија, начини на које се могу предвидети њихове будуће вредности и неки начини избора најбољег времена за њихово одржавање. Читав процес одабира најбољег времена за одржавање састоји се од корака поменутих до сада у редоследу у коме су они поменути. Најпре је потребно учитати скуп података из кога се на сваком кораку врши категоризација и неопходни кораци за ону категорију којој посматрана временска серија припада. Оног тренутка када се утврди да временска серија припада некој категорији за њу се не врше даље провере, на пример за временске серије које имају дневене периоде важи да имају и недељне периоде, те би било могуће доћи у ситуацију да једна временска серија припада више категорија, што не желимо.

Прва провера која се врши је провера да ли је за временску серију могуће изабрати произвољно време за одржавање, односно проверава се степен променљивости вредности посматране временске серије током времена. Након тога се проверава постојање дневних, односно недељних периода на претходно описан начин и за те временске серије врши одабир најбољег времена за оджавање. За преостале временске серије користи се сезонски ARIMA модел. У случају коришћења ARIMA модела се такође разликују два случаја. Један када већ постоје оцењени коефицијенти који се могу употребити за креирање модела и други случај када је те коефицијенте неопходно прво оценити пре креирања модела. Након креирања модела и предвиђања будућих вредности над добијеним предвиђеним вредностима врши се одабир најбољег времена за одржавање коришћењем функције која проналази период са најмањом просечном искоришћености процесора.

Напоменимо још и да постоји одређени број временских серија чије понашање није ни на који начин предвидиво, односно временске серије где није могуће одредити никакав образац који може описати понашање посматране активности процесора. За такве временске серије ниједан од поменутих корака неће дати оптимално решење. У тренутној имплементацији за такве временске серије биће покушано предвиђање будућих вредности коришћењем сезонског ARIMA модела, али предвиђене вредности најчешће неће бити блиске тачним вредностима које ће временска серија имати у будућности, због чега изабрано време одржавања неће бити оптимално. Како би било лакше детектовати такве случајеве може се увести параметар који представља меру сигурности у предвиђене периоде. У том случају би било могуће донети боље

одлуке о томе да ли и како има смисла употребити добијене податке на основу дефинисане мере сигурности у њихову тачност.

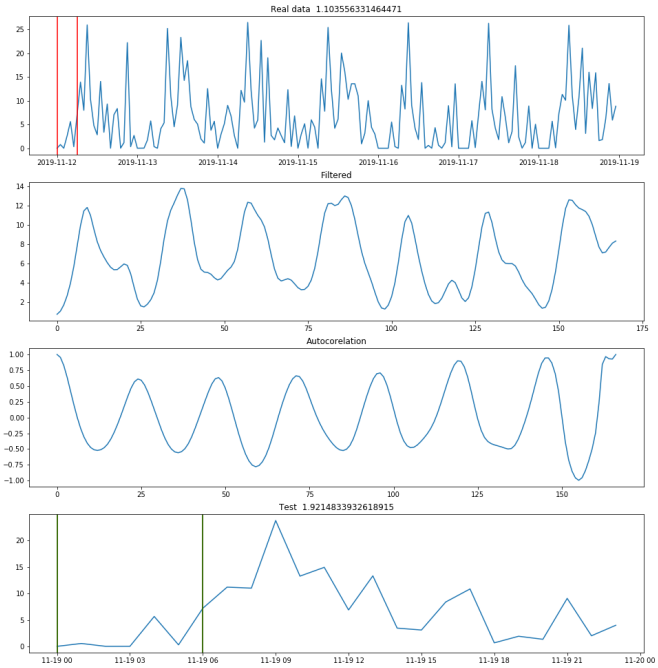
Код 4.4.9

```
for cn,df in data.groupby(['VMName','HostName']):
    df=df[~df.index.duplicated()].asfreq(freq='H')
    if len(df)<168:
        res.append([name,'anytime'])
        continue
    name=df.VMName.unique()[0]
    tz =int(timezones[(timezones.TimeZone == af.get_timezone(df.
                                        ClusterName[0]))].Hour.unique()
                                        [0])
    if af.can_be_updated_anytime(df):
        res.append([name,'anytime'])
        continue
    if af.check_weekly_period(df):
        res.append([cn,name,af.find_n_h(df[(df.index>df.index.max()-
                                            timedelta(days=7))].avg_cpu
                                            ,n,df.index.max(),tz)])
        continue
    if af.check_daily_period(df):
        res.append([cn,name,af.find_n_h(df[(df.index>df.index.max()-
                                            timedelta(days=7))].avg_cpu
                                            ,n,df.index.max(),tz)])
        continue
    if name in has_coefs:
        params=make_tuple(str(coefs[(coefs.VMName==name)].ARIMA_coefs.
                                            unique()[0]))
        model=af.make_model(df.avg_cpu,params)
        predictions=af.make_predictions(model,24)
        res.append([cn,name,af.find_n_h(predictions,test.avg_cpu,6,df.
                                            index.max(),tz)])
    else:
        c=af.arima_coefs(df.avg_cpu)
        coef.append([name,c])
        model=af.make_model(df.avg_cpu,c)
        predictions=af.make_predictions(model,24)
        res.append([cn,name,af.find_n_h(predictions,n,df.index.max(),
                                            tz)])
```

4.5 Резултати

У наставку ће бити приказани графички прикази неких резултата примене претходних корака. Како је опис нових, неактивних или константних временских серија прилично јасан и у случају оваквих временских серија време одржавања сматрамо произвољним, њихов приказ биће изостављен, односно започећемо са приказом временских серија са дневним периодима.

На сликама 4.4 и 4.5 приказане су временске серије са дневним периодима.

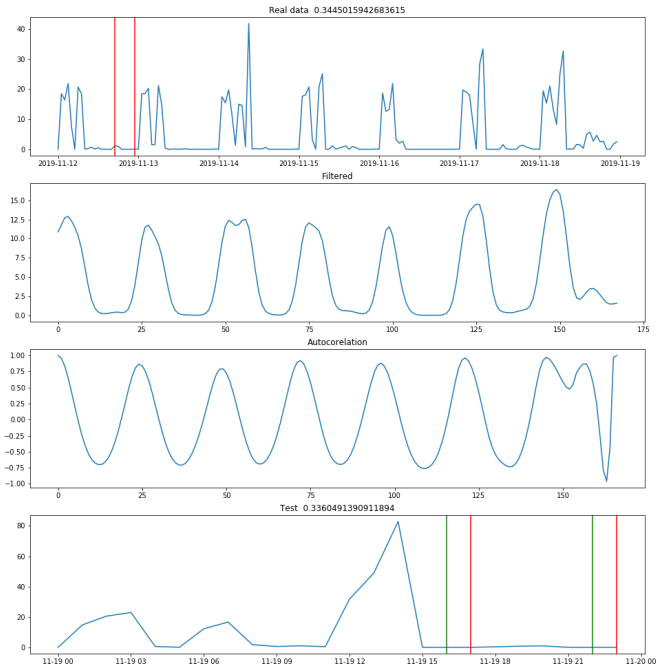


Слика 4.4: Пример временске серије дневним периодом.

На првом графику може се видети временска серија која представља меру активности процесора у последњих недељу дана за које постоје доступни подаци. На том графику црвеном бојом је означен први предвиђени период одржавања како би се приказала његова позиција у времену. Други график приказује филтрирану вредност истог дела временске серије који је приказан на првом графику. Следећи график приказује вредности добијене применом аутокорелационе функције над филтрираним подацима. Коначно, последњи

график приказује првих 24 сати након података над којима је рађено превиђање. Део графика обележен усправним линијама представља изабрано време одржавања и то тако да је време одржавања изабрано коришћењем поступка описаног у поглављу 4.4 означено црвеном бојом, док је оптимално време одржавања означено зеленом бојом.

На обе слике се на првом графику може видети постојање обрасца који се понавља 7 пута, односно понавља се дневно јер се посматрају подаци за 7 дана. Ово потврђују и график на коме су приказане филтриране вредности, односно график који се односи на аутокорелацију на коме се може видети 6 очигледних врхова.



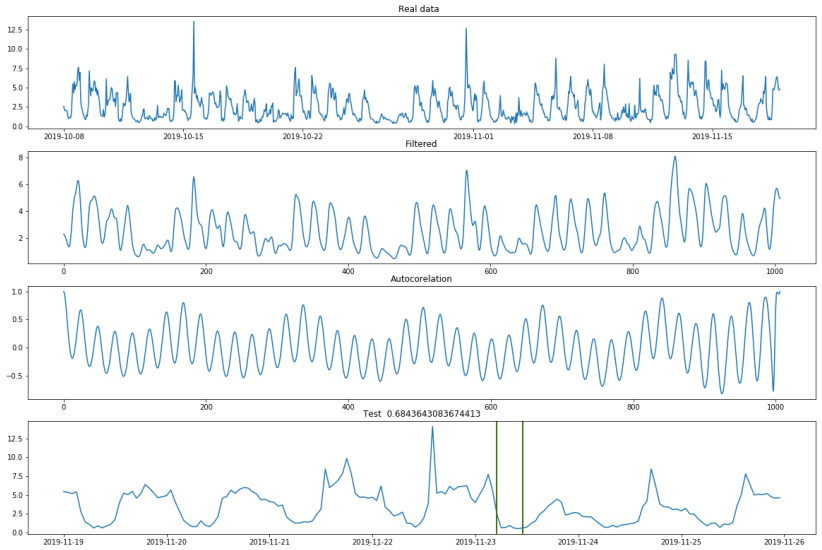
Слика 4.5: Пример временске серије дневним периодом.

На последњем графику приказаном на слици 4.4 постоји само зелена ознака, што говори да су се оптималано и изабрано време одржавања у потпуности поклопили, док на слици 4.5 видимо ситуацију где постоје различито оптимално и изабрано време одржавања. Вредности између црвених и зелених линија овде су веома блиске и преклапају се већим делом. Разлика

између добијених времена је из угла корисника практично занемарљива и представља унапређење у односу на ситуацију када се време одржавања бира насумично у оквиру периода одржавања. Гледајући приказ временске серије на првом графику може се приметити да се узорак коришћења процесора мало изменио током последњег посматраног дана што може бити узрок поменутог одступања.

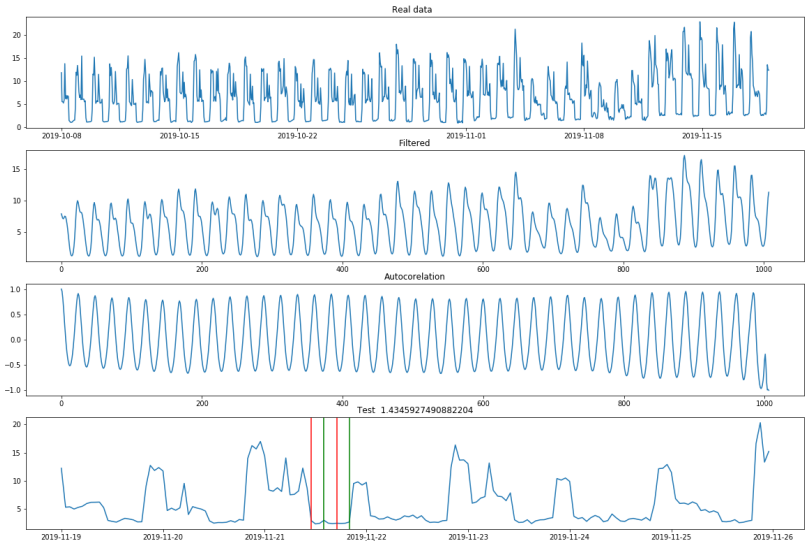
Након дневних периода, на сликама 4.6 и 4.7 погледајмо графичке приказе временских серија које имају недељне периоде.

Значење графика је практично исто као за дневне периоде, уз разлику да се овде приказују подаци за 6 недеља на прва три, односно за недељу дана на последњем графику. Визуелни приказ временске серије потврђује присуство недељних периода, могуће је уочити да се слични обрасци понављају 6 пута, односно сваких недељу дана.



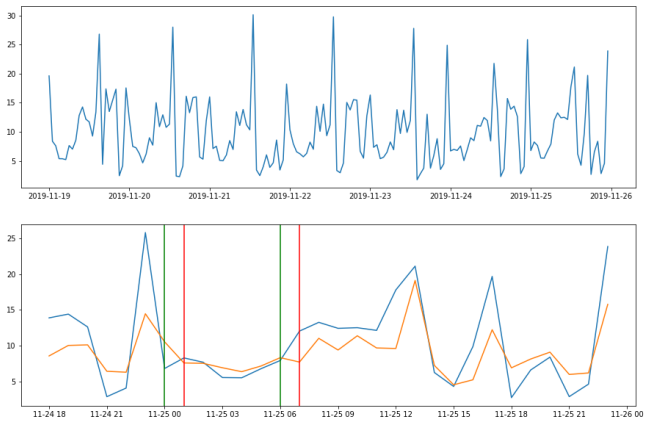
Слика 4.6: Пример временске серије недељним периодом.

На слици 4.6 приказана је временска серија код које су се оптимални и изабрани период поклопили, док је на слици 4.7 приказана временска серија чије се понашање благо изменило током последње посматране недеље, што је узроковало да се оптимални и изабрани период не поклопе у потпуности.



Слика 4.7: Пример временске серије недељним периодом.

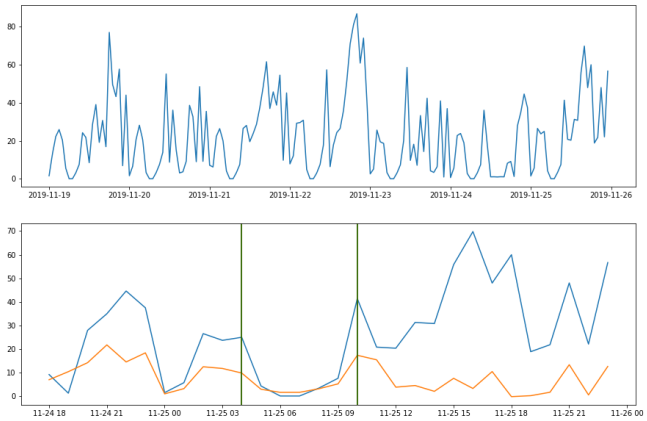
На сликама 4.8, 4.9 и 4.10 приказани су графици временских серија за које је предвиђање вршено ARIMA моделом, након чега је над предвиђеним вредностима изабрано време ажурирања као време када је предвиђена активност процесора минимална.



Слика 4.8: Пример временске серије за коју је коришћен АРИМА модел.

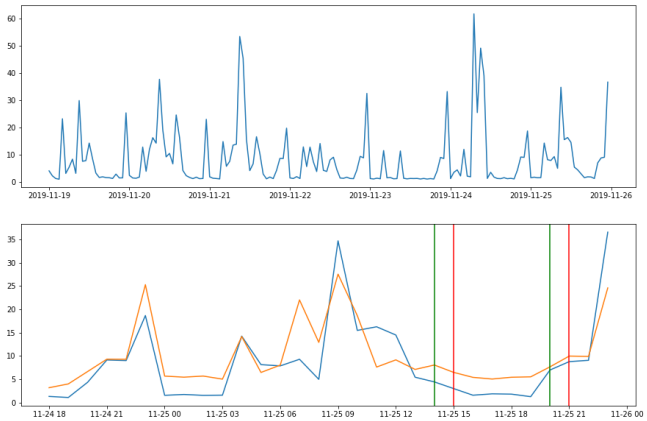
Први график приказује део временске серије који описује податке за последњих недељу дана, док други график приказује вредности за наредни дан, тако да део графика обојен плаво приказује тачне вредности, а део графика

обојен наранџасто вредности добијене као предвиђања модела.



Слика 4.9: Пример временске серије за коју је коришћен АРИМА модел.

Ознаке су и на овим графицима исте као у случају дневних и недељних периода, зелено је означен најбољи период израчунат над тестним подацима, док је црвеном бојом означен период израчунат коришћењем предвиђања модела. Такође, и у овом случају су добијени периоди блиски и у смислу вредности и у смислу броја сати који се преклапају између оптималног и периода изабраног на начин описан у поглављу 4.4.



Слика 4.10: Пример временске серије за коју је коришћен АРИМА модел.

Током израде рада коришћене су различите метрике за рачунање оцене добијених резултата. Коришћење стандардних оцена као што су средњеквадратна грешка (енгл. *Mean Squarred Error, MSE*) и средња апсолутна грешка (енгл. *Mean Absolute Error, MAE*) коришћено је приликом поређења резултата добијених коришћењем различитих конфигурација ARIMA модела и оцењивања потенцијалних унапређења модела приликом изградње модела.

Као једну оцену модела можемо посматрати средњу апсолутну скалирану грешку (енгл. *Mean Absolute Scaled Error, MASE*) чија је вредност за описани модел износила 1.21 и средњу нормализовану корену средњеквадратну грешку (енгл. *Mean Normalized Rooth Mean Squared Error, MNRMSE*) чија је вредност за овај модел износила 0.33.

Такође, коришћењем корака описаних у овом раду за 77% временских серија изабрано време одржавања било је једнако оптималном времену одржавања, док је од преосталих 23% њих 87% имало већа преклапања између изабраног и оптималног времена одржавања, не рачунајући оне временске серије које су означене као оне за које је могуће изабрати произвољан избор времена одржавања.

У 84% случајева средња вредност активности процесора у изабраном временском оквиру се од оптималне разликовала за мање од 5% и била је у прва 3 периода најмање активности процесора у посматраном периоду. Док је у 92% случајева средња вредност у изабраном временском оквиру од оптималне разликовала за мање од 10% и била у првих 5 периода најмање активности процесора.

За око 3% временских серија добијени резултати нису били блиски оптималним и од оптималних су се разликовали за више од 15%. Посматрањем графичких приказа тих временских серија није уочена правилност у њиховом понашању током времена, активност процесора стално се мењала што је утицало да предвиђања не буду прецизна.

Глава 5

Закључак

Примена метода анализе временских серија, омогућава прецизну анализу података о активности процесора у окружењу облака током времена. Описане методе анализе временских серија омогућавају откривање периода најмање активности процесора, који представљају најбоље време за извођење одржавања и управљања ресурсима без значајних прекида и утицаја на кориснике. Одабир оптималног времена за одржавање има значајне практичне предности у окружењу облака и доприноси побољшаној доступности и перформансама система.

Периоди најмање активности процесора могу бити искоришћени за одржавање и управљање ресурсима, што има потенцијал да значајно смањи трошкове и побољша перформансе у облаку. Компаније и пружаоци услуга у облаку могу аутоматски планирати периоде одржавања и других операција у периоду времена када је оптерећење процесора најниже, што смањује утицај на кориснике и омогућава ефикасније управљање ресурсима. Добијени резултати могу се уз мале измене уопштити за примену у другим областима оптимизације управљања ресурсима, на пример могу се користити за аутоматско скалирање ресурса на основу тренутних потреба, што може довести до ефикаснијег и економичнијег коришћења ресурса, као и смањења трошкова.

Идеја употребе добијених решења је креирање извршног фајла који ће радити предвиђања за све временске серије које буду прослеђене као улаз и који ће затим добијене резултате аутоматски уписивати на места која дефинишу када је могуће радити операције које имају утицај на доступност или перформансе корисничких ресурса. Тада ће се у случају да је потребно извршити неку такву операцију проверити да ли већ постоји дефинисано време одржа-

вања, као и да ли је његова дужина довољна за извршавање дате операције. Уколико такво време постоји дата операција биће заказана за извршавање у пронађеном времену одржавања, а у супротном ће се покренути процес који ће израчунати ново време одржавања чија ће дужина бити прилагођена тренутним потребама.

Како рачунање најбољег времена за одржавање захтева одређено време потребно за извршавање, има смисла дефинисати апликацију на нивоу центра података која би покренула процес израчунавања најбољих времена за одржавање почетком сваке недеље за текућу недељу. Резултати би били уписани на одговарајуће локације и могли бити коришћени по потреби. Тада би потребни подаци о најбољем времену за одржавање у највећем броју случајева већ били доступни у тренутку када је потребно користити ту информацију.

Рачунарство у облаку се све више развија, а са његовим развојем има и све више захтева које је потребно испунити како би пружалац услуга у облаку био конкурентан и имао понуду која је корисницима привлачнија од понуде конкурента. Због тога се у рачунарству у облаку стално ради на различитим унапређењима која би повећала задовољство корисника услугом. Примена одабира времена одржавања на паметан начин изложена у овом раду може бити веома примамљива за кориснике јер би се прекиди дешавали у тренуцима у којима би имали најмањи утицај на њихово пословање.

Библиографија

[1] Amazon AWS. on-line at: `https://aws.amazon.com/`, accessed: 06-2023.

[2] Availability sets overview. on-line at `https://learn.microsoft.com/en-us/azure/virtual-machines/availability-set-overview`, accessed: 06-2023.

[3] matplotlib. on-line at: `https://matplotlib.org/`, accessed: 06-2023.

[4] Microsoft Azure. on-line at: `https://aws.amazon.com/`, accessed: 06-2023.

[5] NumPy. on-line at: `https://numpy.org/`, accessed: 06-2023.

[6] pandas. on-line at: `https://pandas.pydata.org/`, accessed: 06-2023.

[7] pmdarima. on-line at: `https://pypi.org/project/pmdarima/`, accessed: 06-2023.

[8] SciPy. on-line at: `https://scipy.org/`, accessed: 06-2023.

[9] statsmodels. on-line at: `https://www.statsmodels.org/`, accessed: 06-2023.

[10] Peter J. Brockwell and Richard A.Davis. *Introduction to Time Series and Forecasting*. Springer International Publishing, 2016.

[11] Rodrigo N. Calheiros, Enayat Masoumi, Rajiv Ranjan, and Rajkumar Buyya. Workload prediction using arima model and its impact on cloud applications' qos. *IEEE Transactions on Cloud Computing*, 2015.

[12] Wiliam Y. Chang, Hosame Abu-Amara, and Jessica Feng Sanford. *Transforming Enterprise Cloud Services*. Springer International Publishing, 2010.

[13] Eli Cortez, Anand Bonde, Alexandre Muzio, Mark Russinovich, Marcus Fontoura, and Ricardo Bianchini. Resource central: Understanding and predicting workloads for improved resource management in large cloud platforms. *USENIX Annual Technical Conference*, 2017.

[14] Arijit Khan, Xifeng Yan, Shu Tao, and Nikos Anerousis. Workload characterization and prediction in the cloud: A multiple time series approach. In *2012 IEEE Network Operations and Management Symposium*, 2012.

[15] John McCarthy. Reminiscences on the history of time sharing. *IEEE Annals of the History of Computing, Volume 14, Issue 1*, 1992.

[16] Peter Mell and Tim Grance. The nist definition of cloud computing. *Recommendations of the National Institute of Standards and Technology*, 2011.

[17] Iulian Neamtiu and Tudor Dumitras. Cloud software upgrades: Challenges and opportunities. *International Workshop on the Maintenance and Evolution of Service-Oriented and Cloud-Based Systems, MESOCA*, 2011.

[18] Isaac Odun-Ayo, Olasupo Ajayi, and Nicholas Omoregbe. Cloud service level agreements –issues and development. *Advances in Science, Technology and Engineering Systems Journal*, 2017.

[19] Pankesh Patel, Ajith Ranabahu, and Amit Sheth. Service level agreement in cloud computing. *Cloud Workshops at OOPSLA09*, 2009.

[20] Ruey S. Tsay. *Analysis of Financial Time Series*. University of Chicago, Graduate School of Business, 2005.

[21] Весна Јевремовић Јован Малишић. *Случајни ūроцеси и временске серије*. Математички факултет у Београду, 2008.

[22] Александра Нојковић Зорица Младеновић. *Примењена анализа временских серија*. Центар за издавачку делатност Економског факултета у Београду, 2012.

[23] Златко Ј. Ковачић. *Анализа временских серија*. Економски факултет у Београду, 1995.

[24] Бојана Милошевић. *Основи статистике*. Математички факултет у Београду, 2021.

[25] Павле Младеновић. *Вероватноћа и статистика*. Математички факултет у Београду, 2008.

Биографија аутора

Наталија Драшковић рођена је 02.05.1997. у Лесковцу. Основну школу „Јосиф Костић" завршила је 2012. године као носилац Вукове дипломе и ученик генерације. Природно-математички смер Гимназије у Лесковцу завршила је 2016. године, такође као носилац Вукове дипломе. Основне академске студије уписала је исте године на смеру Рачунарство и информатика на Математичком факултету. На наведеном смеру дипломирала је 2020. године након чега је на истом смеру уписала и мастер академске студије. Током последње године основних студија, од октобра 2019. одрадила је праксу на позицији софтвер инжењера (енгл. *Software Engineer*) у Мајкрософт развојном центру Србија (енгл. Microsoft Development Center Serbia), где је од јуна 2020. и запослена на позицији софтвер инжењера.