Szakdolgozat

Miskolci Egyetem

Rutinszerű feladatok automatizálása grafikus felhasználói felületek esetében

Készítette:

Pázmándi Erik

Programtervező informatikus

Témavezető:

Dr. Kovács Béla

Konzulens:

Piller Imre

Miskolc, 2022

Miskolci Egyetem
Gépészmérnöki és Informatikai Kar
Alkalmazott Matematikai Intézeti Tanszék **Szám:**

Szakdolgozat Feladat

Pázmándi Erik (GXN833) programtervező informatikus jelölt részére.

A szakdolgozat tárgyköre: Folyamatelemzés, RPA

A szakdolgozat címe: Rutinszerű feladatok automatizálása grafikus felhasználói felületek esetében

A feladat részletezése:

A számítógépek kifejlesztésének és használatának egyik fő motivációja, hogy a segítségével az automatizált módon végrehajtható folyamatok emberi beavatkozás nélkül is végrehajthatóak legyenek. Ennek ellenére számos esetben tapasztalhatjuk, hogy az alkalmazások felhasználói felületén rutinszerűen, repetitíven hajtanak végre műveleteket.

A dolgozat azt vizsgálja, hogy ezek a folyamatok a korábban rögzített eseménysorok alapján hogyan ismerhetők fel. Bemutatja az RPA (Robotic Process Automation) eszközkészletét, többek között a folyamatelemzés elterjedt módszereit, alkalmazási lehetőségeit, a grafikus felhasználói felületekhez kapcsolódó speciális eseteket. Az elemzésekhez, automatizálást segítő eszköz elkészítéséhez Microsoft Windows platformon Delphi programozási nyelv kerül felhasználásra.

Témavezető: Dr. Kovács Béla (egyetemi docens)

Konzulens: Piller Imre (egyetemi tanársegéd)

A feladat kiadásának ideje: 2021. Szeptember 23.

. . . . . .
szakfelelős

Eredetiségi Nyilatkozat

Alulírott **Pázmándi Erik**; Neptun-kód: `GXN833` a Miskolci Egyetem Gépészmérnöki és Informatikai Karának végzős Programtervező informatikus szakos hallgatója ezennel büntetőjogi és fegyelmi felelősségem tudatában nyilatkozom és aláírásommal igazolom, hogy *Rutinszerű feladatok automatizálása grafikus felhasználói felületek esetében* című szakdolgozatom saját, önálló munkám; az abban hivatkozott szakirodalom felhasználása a forráskezelés szabályai szerint történt.

Tudomásul veszem, hogy szakdolgozat esetén plágiumnak számít:

- szószerinti idézet közlése idézőjel és hivatkozás megjelölése nélkül;
- tartalmi idézet hivatkozás megjelölése nélkül;
- más publikált gondolatainak saját gondolatként való feltüntetése.

Alulírott kijelentem, hogy a plágium fogalmát megismertem, és tudomásul veszem, hogy plágium esetén szakdolgozatom visszautasításra kerül.

Miskolc, ...év ..... .hó ..... .nap

.....

Hallgató

1.

szükséges (módosítás külön lapon)

A szakdolgozat feladat módosítása

nem szükséges

....

dátum témavezető(k)

2. A feladat kidolgozását ellenőriztem:

témavezető (dátum, aláírás): konzulens (dátum, aláírás):

....

....

....

3. A szakdolgozat beadható:

....

dátum témavezető(k)

4. A szakdolgozat ..... szövegoldalt

..... program protokollt (listát, felhasználói leírást)

..... elektronikus adathordozót (részletezve)

.....

..... egyéb mellékletet (részletezve)

.....

tartalmaz.

....

dátum témavezető(k)

5.

bocsátható

A szakdolgozat bírálatra

nem bocsátható

A bíráló neve: .....

....

dátum szakfelelős

6. A szakdolgozat osztályzata

a témavezető javaslata: .....

a bíráló javaslata: .....

a szakdolgozat végleges eredménye: .....

Miskolc, ...

a Záróvizsga Bizottság Elnöke

Tartalomjegyzék

1. fejezet

Bevezetés

A dolgozat azt vizsgálja, hogy a számítógépek felhasználói felületén miféle sokszor elismételt folyamatok zajlanak le, ezek egy robot szempontjából hogyan is néznek ki, hogyan lehet ezeket utánozni, illetve automatizálni. Bemutat folyamatelemzési módszereket, ... BŐVÍTENI! (1-2 oldal)

2. fejezet

Folyamatelemzés

2.1. Alpha-algoritmus

Forrás: (Alpha-algoritmus, 2022).

Az alpha algoritmus (vagy alpha bányász) mint folyamatelemzési algoritmus célja, hogy eseménysorozatok halmazából egy ok-okozat rendszert építsen fel. Először van der Aalst, Weijters és Măruşter hozta be a köztudatba. A működésében az eseménysorok halmazát nevezhetjük eseménynaplónak is. Ez az eseménynapló úgynevezett trace halmazoknak a halmaza, egy trace pedig adott tevékenységnek a sorozata.

Az alpha bányász volt a legelső folyamatbányászati módszer amit valaha javasoltak és egy egész jó rálátást biztosít a folyamatbányászat céljára, valamint arra, hogy a folyamatokban lévő különböző tevékenységek hogyan is vannak végrehajtva. Emelett, az alpha bányász szolgált számos újabb folyamatbányászati technika (pl.: Heurisztikus bányász, genetikus bányászat) alapjaként.

2.1.1. Rövid leírása

Az algoritmus egy munkafolyamati naplót $W \subseteq T^*$ kap bemenetként, és eredményként egy munkafolyamati hálót épít fel.

Ezt az alapján csinálja meg, hogy megvizsgálja az általános kapcsolatokat az egyes feladatok között. Például egy adott feladat lehet, hogy minden esetben megelőz egy másik feladatot, ami egy hasznos információ.

Definíciók

- *Munkafolyamati trace*: Egy string a T ábécé feladatai közül.
- *Munkafolyamati napló*: Munkafolyamati tracek halmaza.

2.1.2. Eseménynapló

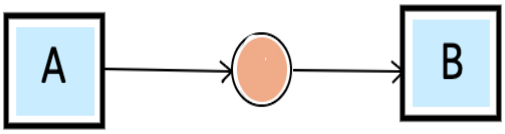
Az eseménynapló az elsődleges szükséglet bármely folyamatbányászati algoritmus alkalmazásához. Az eseménynapló a következőket tartalmazza: egyedi azonosító az esethez, tevékenység megnevezése valamint egy időbélyeg. Egy eseménynaplót akár tevékenységek halmazának halmazaként is lehet ábrázolni.

Az alpha bányász szabályai szerint az egyes tevékenységek között az alábbi 4 féle kapcsolat egyike lehetséges:

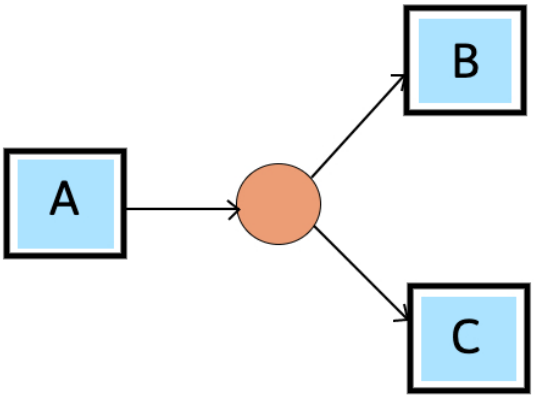
- **Közvetlen sorrend:** $x > y$ akkor és csakis akkor ha az x eseményt közvetlenül követi y.
- **Okozat:** $x \to y$ ha $x > y$ és nem $y > x$.
- **Párhuzam:** $x \parallel y$ ha $x > y$ és $y > x$.
- **Választás:** $x \# y$ ha nem $(x > y)$ és nem $(y > x)$.

2.1.3. Minták

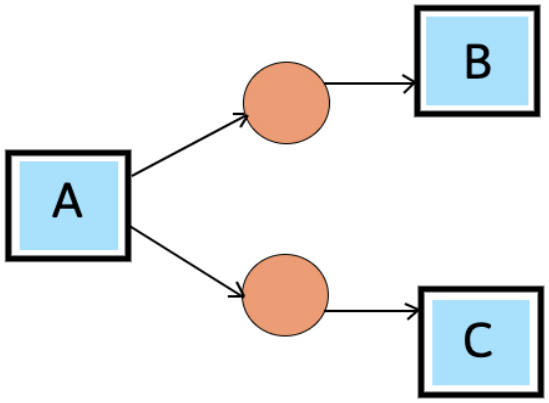
1. **Szekvencia: A → B**



2. **XOR-elágazás: A → B, A → C és B # C**



3. **ÉS-elágazás: A → B, A → C és B ∥ C**



2.1.4. Példa

Vegyük példának a következő eseménynaplót:

ID	Tevékenység	Időbélyeg
1	A	2022-10-05 13:50:40.000
1	B	2022-10-05 16:30:12.000
1	C	2022-10-05 16:57:31.000
1	D	2022-10-06 13:50:41.000
2	A	2022-10-06 15:30:27.000
2	C	2022-10-06 16:23:33.000
2	B	2022-10-07 08:33:02.000
2	D	2022-10-07 12:41:11.000
3	A	2022-10-07 13:02:57.000
3	E	2022-10-07 14:11:21.000
3	D	2022-10-07 14:59:22.000

Ebben az esetben az eseménynaplót az alábbi módon tudjuk jelölni:

$$L_1 = [< A, B, C, D >, < A, C, B, D >, < A, E, D >]$$

Az alpha bányász úgy kezdi a munkát, hogy az eseménynaplót közvetlen-sorrend, okozat, párhuzam és választás relációkra alakítja és ezeket felhasználva létrehoz egy petri hálót ami leírja a folyamat modellét.

Első lépésként létrehoz egy lenyomati mátrixot:

	a	b	c	d	e
a	#	$\rightarrow$	$\rightarrow$	#	$\rightarrow$
b	$\leftarrow$	#	$\|\|$	$\rightarrow$	#
c	$\leftarrow$	$\|\|$	#	$\rightarrow$	#
d	#	$\leftarrow$	$\leftarrow$	#	$\leftarrow$
e	$\leftarrow$	#	#	$\rightarrow$	#

Y_W az összes (A, B) pár halmaza a feladatok maximális halmazából úgy, hogy:

- Egyik $A \times A$ és $B \times B$ sem tagja $>$-nek, és
- $A \times B$ részhalmaza $\rightarrow$-nek.

P_W tartalmazza az egyes Y_W-hez tartozó helyeket $p_(A, B)$, plussz a beviteli i_W helyet és a kimeneti o_W helyet.

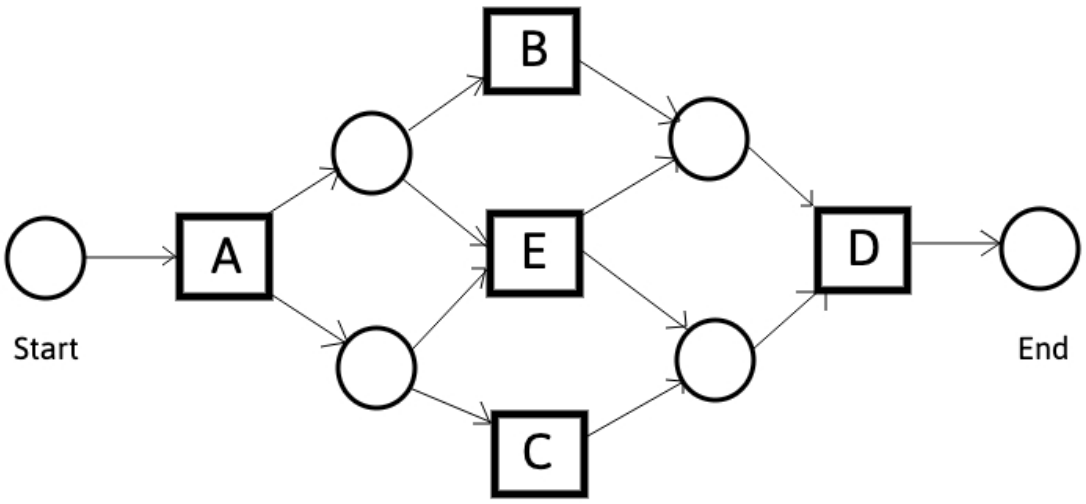
A folyamati reláció F_W az alábbiak uniójából áll össze:

- $\{(a, p_{(C,B)})|(A, B) \in Y_W \wedge a \in A\}$
- $\{(p_{(A,B)}, b)|(A, B) \in Y_W \wedge b \in B\}$
- $\{(i_W, t)|t \in T_1\}$
- $\{(t, i_0)|t \in T_0\}$

Az eredmény

- egy Petri háló struktúra $\alpha(W) = (P_W, T_W, F_W)$
- egy beviteli hellyel i_W és egy kimeneti hellyel o_W
- mivel minden T_W átmenet F_W-úton van i_W-ből o_W-be, így valóban egy munkafolyamati háló.

Ehhez a példához az alábbi petri háló jönne létre az alpha bányász használatával:



2.1.5. Korlátozások

- **Implicit helyek**: Az alpha bányász nem tud különbséget tenni az implicit és a szükséges helyek között, így a felfedezett petri hálóban előfordulhatnak plusz szükségetelen helyek.
- **Ciklusok**: Az alpha bányász nem képes 1-gyes és 2-tes hosszúságú ciklusok felismerésére a folyamatmodellben.
- A helyi függőségeket gyakran nem veszi észre az alpha bányász.

3. fejezet

Tervezés

Itt kezdődik a dolgozat lényegi része, úgy értve, hogy a saját munka bemutatása. Jellemzően ebben szerepelni szoktak blokkdiagramok, a program struktúrájával foglalkozó leírások. Ehhez célszerű UML ábrákat (például osztály- és szekvenciadiagramokat) használni.

Amennyiben a dolgozat inkább kutatás jellegű, úgy itt lehet konkretizálni a kutatási módszertant, a kutatás tervezett lépéseit, az indoklást, hogy mit, miért és miért pont úgy érdemes csinálni, ahogyan az a későbbiekben majd részletezésre kerül.

Ebben a fejezetben az implementáció nem kell, hogy túl nagy szerepet kapjon. Ez még csak a tervezési fázis. (Nyilván ha olyan a téma, hogy magának az implementációnak a módjával foglalkozik, adott formális nyelvet mutat be, úgy a kódpéldákat már innen sem lehet kihagyni.)

3.1. Táblázatok

Táblázatokhoz a `table` környezetet ajánlott használni. Erre egy minta a 3.1. táblázat. A hivatkozáshoz az egyedi `label` értéke konvenció szerint `tab:` prefixszel kezdődik.

3.1. táblázat. Minta táblázat. A táblázat felirata a táblázat felett kell legyen!

a	b	c
1	2	3
4	5	6

3.2. Ábrák

Ábrákat a `figure` környezettel lehet használni. A használatára egy példa a 3.1. ábrán látható. Az `includegraphics` parancsba Az ábrák felirata az ábra alatt kell legyen. Az ábrák hivatkozásához használt nevet konvenció szerint `fig:`-el célszerű kezdeni.

3.3. További környezetek

A matematikai témájú dolgozatokban szükség lehet tételek és bizonyításaik megadására. Ehhez szintén vannak készen elérhető környezetek.

3.1. ábra. A Miskolci Egyetem címere.

3.1. definíció. Ez egy definíció

3.2. lemma. *Ez egy lemma*

3.3. tétel. *Ez egy tétel*

Bizonyítás. Ez egy bizonyítás □

3.4. következmény. *Ez egy tétel*

3.5. megjegyzés. Ez egy megjegyzés

3.6. példa. Ez egy példa

4. fejezet

Megvalósítás

Ez a fejezet mutatja be a megvalósítás lépéseit. Itt lehet az esetlegesen előforduló technikai nehézségeket említeni. Be lehet már mutatni a program elkészült részeit.

Meg lehet mutatni az elkészített programkód érdekesebb részeit. (Az érdekesebb részek bemutatására kellene szorítkozni. Többségében a szöveges leírásnak kellene benne lennie. Abból lehet kiindulni, hogy a forráskód a dolgozathoz elérhető, azt nem kell magába a dolgozatba bemásolni, elegendő csak behivatkozni.)

A dolgozatban szereplő forráskódrészletekhez külön vannak programnyelvenként stílusok. Python esetében például így néz ki egy formázott kódrészlet.

```
import sys

if __name__ == '__main__':
    pass
```

A stílusfájlok a `styles` jegyzékben találhatók. A stílusok között szerepel még C++, Java és Rust stílusfájl. Ezek használatához a `dolgozat.tex` fájl elején `usepackage` paranccsal hozzá kell adni a stílust, majd a stílusfájl nevével megegyező környezetet lehet használni. További példaként C++ forráskód esetében ez így szerepel.

```
#include <iostream>

class Sample : public Object
{
    // An empty class definition
}
```

Stílusfájlokból elegendő csak annyit meghagyni, amennyire a dolgozatban szükség van. Más, C szintaktikájú nyelvekhez (mint például a JavaScript és C#) a Java vagy C++ stílusfájlok átszerkesztésére van szükség. (Elegendő lehet csak a fájlnevet átírni, és a fájlban a környezet nevét.)

Nyers adatok, parancssori kimenetek megjelenítéséhez a `verbatim` környezetet lehet használni.

```
$ some commands with arguments
1 2 3 4 5
$ _
```

A kutatás jellegű témáknál ez a fejezet gyakorlatilag kimaradhat. Helyette inkább a fő vizsgálati módszerek, kutatási irányok kaphatnak külön-külön fejezeteket.

5. fejezet

Tesztelés

A fejezetben be kell mutatni, hogy az elkészült alkalmazás hogyan használható. (Az, hogy hogyan kell, hogy működjön, és hogy hogy lett elkészítve, az előző fejezetekben már megtörtént.)

Jellemzően az alábbi dolgok kerülhetnek ide.

- Tesztfuttatások. Le lehet írni a futási időket, memória és tárigényt.
- Felhasználói kézikönyv jellegű leírás. Kifejezetten a végfelhasználó szempontjából lehet azt bemutatni, hogy mit hogy lehet majd használni.
- Kutatás kapcsán ide főként táblázatok, görbék és egyéb részletes összesítések kerülhetnek.

6. fejezet

Összefoglalás

Hasonló szerepe van, mint a bevezetésnek. Itt már múltidőben lehet beszélni. A szerző saját meglátása szerint kell összegezni és értékelni a dolgozat fontosabb eredményeit. Meg lehet benne említeni, hogy mi az ami jobban, mi az ami kevésbé jobban sikerült a tervezettnél. El lehet benne mondani, hogy milyen további tervek, fejlesztési lehetőségek vannak még a témával kapcsolatban.

Irodalomjegyzék

Alpha-algoritmus. (2022). *Alpha-algoritmus — Wikipedia, the free encyclopedia.* Retrieved from `https://en.wikipedia.org/wiki/Alpha_algorithm` ([Online, 2022-Október-07])

CD Használati útmutató

Ennek a címe lehet például *A mellékelt CD tartalma* vagy *Adathordozó használati útmutató* is.

Ez jellemzően csak egy fél-egy oldalas leírás. Arra szolgál, hogy ha valaki kézhez kapja a szakdolgozathoz tartozó CD-t, akkor tudja, hogy mi hol van rajta. Jellemzően elég csak felsorolni, hogy milyen jegyzékek vannak, és azokban mi található. Az elkészített programok telepítéséhez, futtatásához tartozó instrukciók kerülhetnek ide.

A CD lemezre mindenképpen rá kell tenni

- a dolgozatot egy `dolgozat.pdf` fájl formájában,
- a LaTeX forráskódját a dolgozatnak,
- az elkészített programot, fontosabb futási eredményeket (például ha kép a kimenet),
- egy útmutatót a CD használatához (ami lehet ez a fejezet külön PDF-be vagy MarkDown fájlként kimentve).