mgr inż. Damian Giebas

Metoda wykrywania konfliktów zasobowych w aplikacjach wielowątkowych

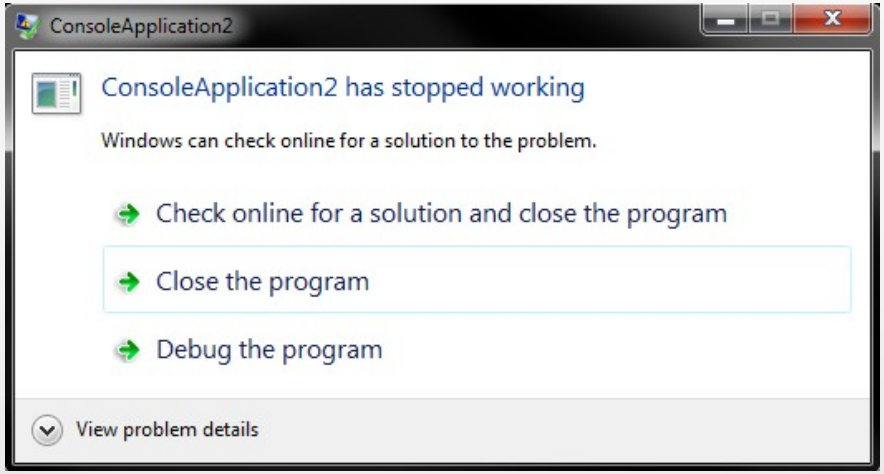
Opiekun naukowy: prof. dr hab. inż. Grzegorz Bocewicz

Agenda

- Motywacja
- Stosowane modele, wykrywanie, zapobieganie i naprawa konfliktów zasobowych
- Przedstawienie problemu i omówienie metody
- Podsumowanie

Motywacja

- Powszechność aplikacji wielowątkowych.
- Użytkownik aplikacji wielowątkowej może doświadczyć:
  - niepoprawnego działania aplikacji
  - zawieszania pracy aplikacji
  - nieoczekiwanego zakończenia pracy aplikacji



Motywacja

Aplikacje, w których może dojść do konfliktów zasobowych tworzone są m. in. za pomocą języków:

Lp.	Język programowania	Popularność na świecie
1.	Java	14,880%
2.	C	13,305%
3.	Python	8,262%
4,	C++	8,126%
5.	Visual Basic .NET	6,429%
6.	C#	3,267%
...	...	...
35.	**Rust**	**0,310%**

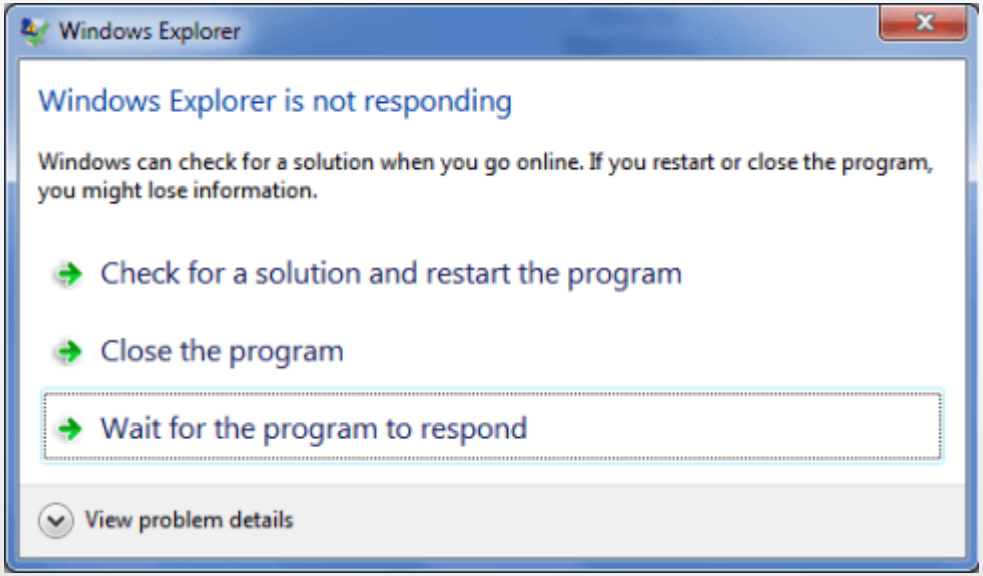
Lista stworzona w oparciu o ranking TIOBE z dnia 16 Marca 2019.

Zasoby i Konflikt zasobowy

	Wątek 1	Wątek 2	Zasób
Krok 1	Odczyt Pamięć: 0		0
Krok 2	Modyfikacja Pamięć: 1		0
Krok 3	Zapis Pamięć: 1		1
Krok 4		Odczyt Pamięć: 1	1
Krok 5		Modyfikacja Pamięć: 2	1
Krok 6		Zapis Pamięć: 2	2

Blokady i zmienne warunkowe

- Blokady (ang. mutex) – mechanizm synchronizacji gwarantujący wzajemne wykluczanie się
- Zmienne warunkowe (ang. condition variable) – mechanizm komunikacji między wątkami. Zmienne warunkowe zawsze wykorzystują blokady.



Skutki konfliktów zasobowych

Szkodliwa rywalizacja (ang. race condition)

- 171 prac naukowych

Zakleszczenie (ang. deadlock)

- 1409 prac naukowych

Naruszenie niepodzielności (ang. atomicity violation)

- 18 prac naukowych

Naruszenie porządku (ang. order violation)

- 25 prac naukowych

Stan prac pochodzi z bazy Scopus z dnia 23 marca 2019 roku i ogranicza się do dziedzin Informatyki i Inżynierii.

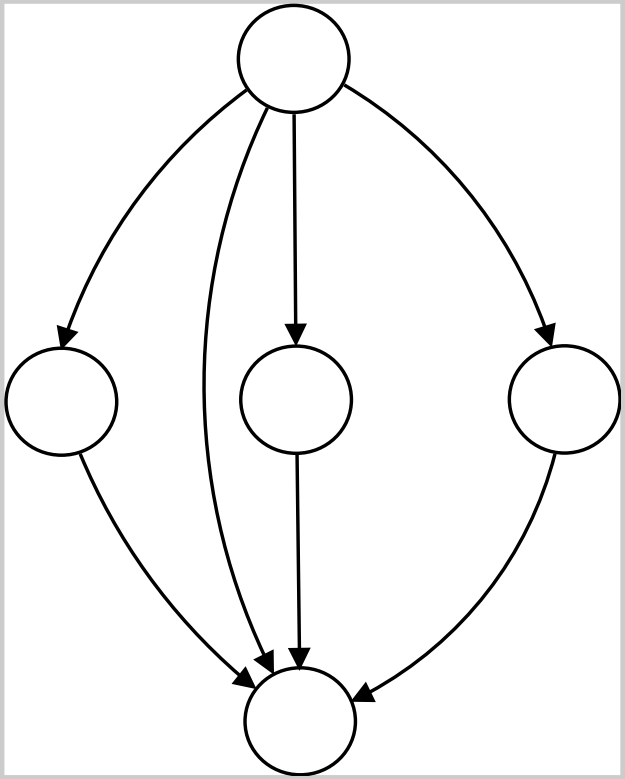
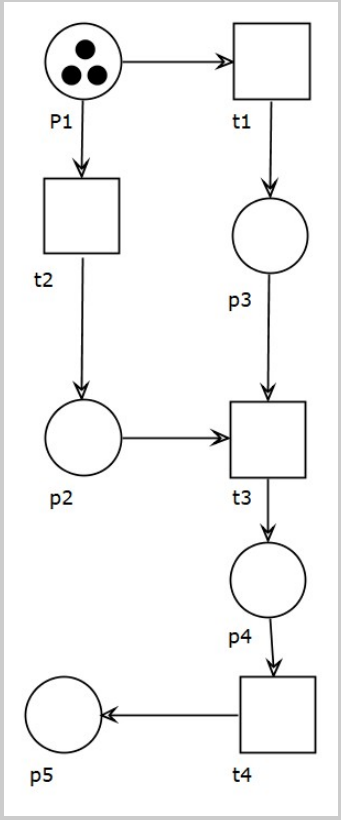
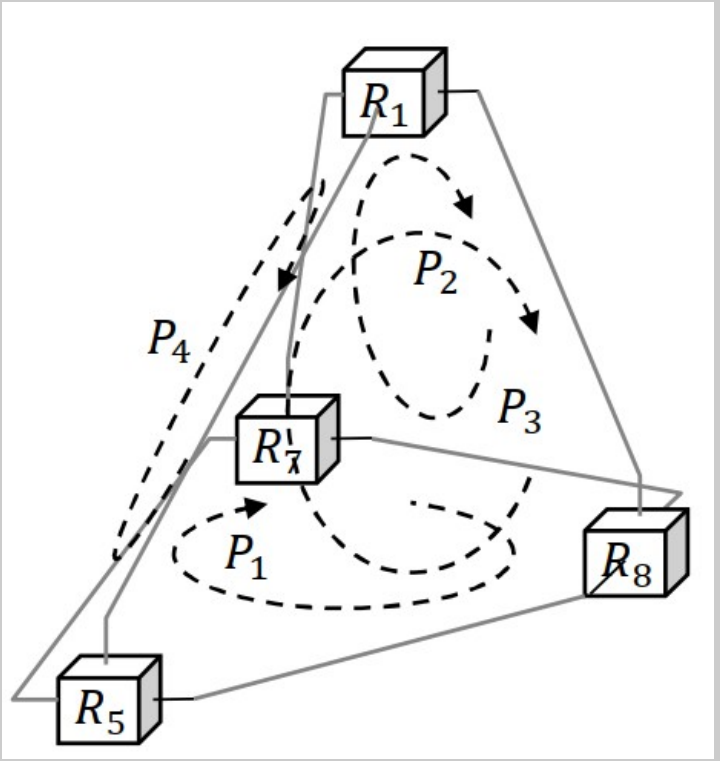
Motywacja

- Konflikty zasobowe mogą występują między innymi:
  - w naturze
  - w systemach komunikacji
  - w elektronice
  - w systemach rozproszonych

Przykład zakleszczenia w naturze.
Źródło: http://www.bweinh.com

Zakleszczenie w komunikacji.
Źródło: http://www.wikiwand.com

Modele matematyczne

Graf Przepływu Sterowania	Sieci Petriego	Systemy Współbieżnych Procesów
		

D. Giebas, R. Wojszczyk, „Graphical representations of multithreaded applications”, Applied Computer Science, 14(2), Politechnika Lubelska, strony 20-37, Lublin 2018

Wykrywanie konfliktów zasobowych

- Narzędzia wspomagające wykrywanie:
  - RacerX
  - Relay
  - Ctrigger
  - Eraser
  - ThreadSanitizer
  - Helgrind
- Testy obciążeniowe
- Inspekcja kodu (ang. code review)

Zapobieganie konfliktom i ich naprawa

- Frameworki i rozszerzenia:
  - Charm++
  - Intel Threading Building Blocks
  - OpenMP
  - Cilk
- Narzędzie usuwające wielowątkowość:
  - Grace
- Narzędzia usuwające wybrane konflikty:
  - Axis
  - AFix

Problem

Dane:

- Kod aplikacji napisany w języku C
- Biblioteka pthread

Pytania:

- Czy w aplikacji występują konflikty zasobowe?
- Gdzie znajdują się konflikty zasobowe w kodzie aplikacji?

Ograniczenia:

- Składnia i semantyka języka C
- Programowanie strukturalne
- Minimum dwa wątki w aplikacji
- Konflikty powodujące zjawiska:
  - Szkodliwa rywalizacja
  - Zakleszczenie
  - Naruszenie niepodzielności
  - Naruszenie porządku

Wizualizacja metody

Kod źródłowy

Metoda

Aplikacja zawiera konflikty zasobowe

Aplikacja nie zawiera konfliktów zasobowych

Metoda

Etap I

- Transformacja kodu źródłowego

Etap II

- Wykrywanie konfliktów zasobowych powodujących:
  - szkodliwą rywalizację
  - zakleszczenia
  - naruszenie niepodzielności
  - naruszenie porządku

Etap III

- Wygenerowanie raportu

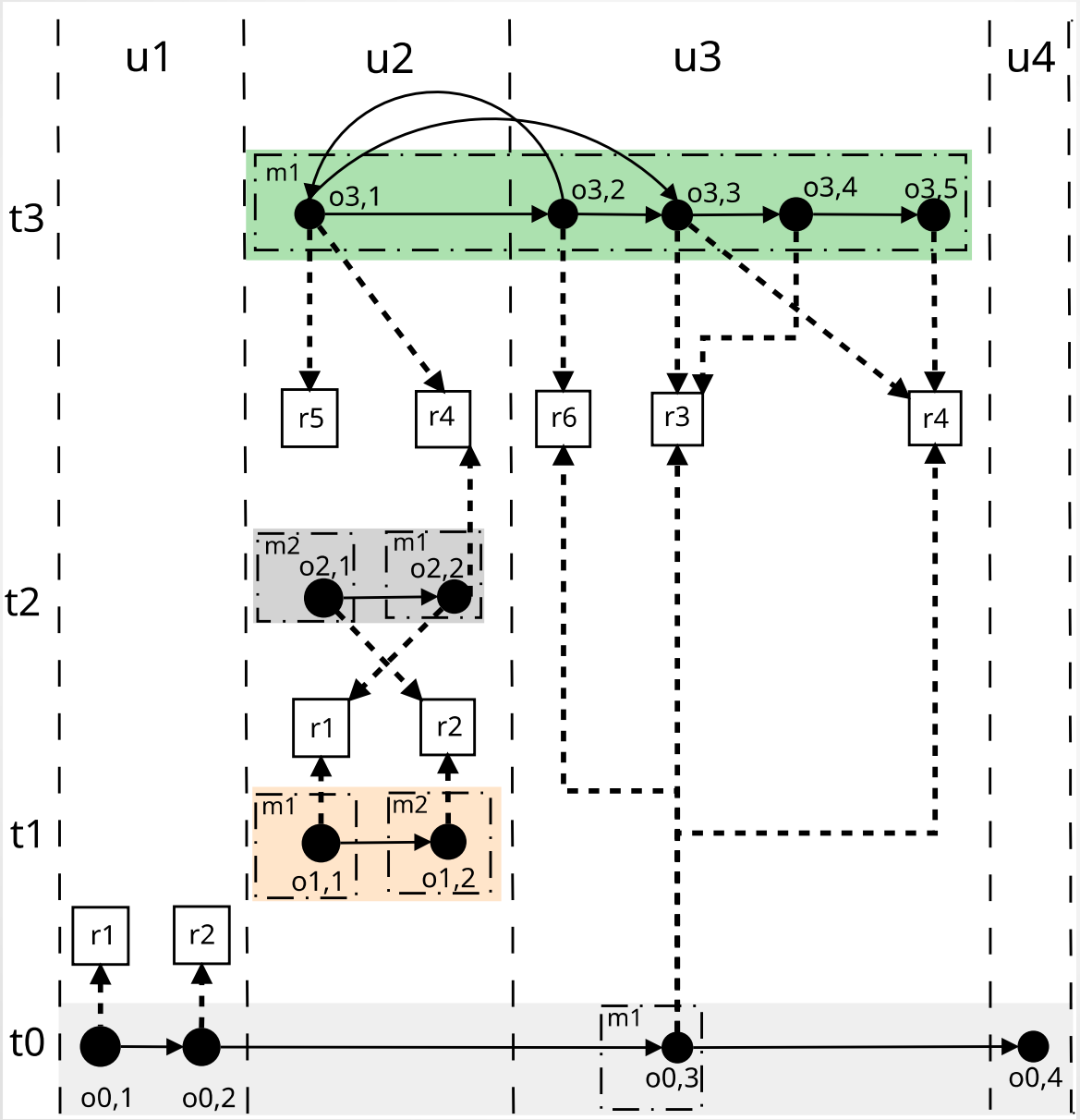
Model aplikacji wielowątkowej

$$C_P = (T_P, U_P, R_P, O_P, S_P, M_P, F_P)$$

Gdzie:
P – indeks aplikacji
T_P – zbiór wątków
U_P – sekwencja zbiorów przedziałów czasu
R_P – zbiór zasobów współdzielonych
O_P – zbiór wszystkich operacji
S_P – zbiór operacji chronionych blokadą
M_P – sekwencja blokad
F_P – zbiór krawędzi obejmujący:

- krawędzie przejścia
- krawędzie użycia
- krawędzie zależności

Reprezentacja aplikacji za pomocą modelu



Teza

Metoda statycznej analizy kodu aplikacji pozwala na wykrywanie konfliktów zasobowych w aplikacjach wielowątkowych.

Aktualne osiągnięcia

Twierdzenie 1 *Niech* $O_P = \{o_{m,j}, ..., o_{n,q}\}$ *oznacza zbiór operacji realizowanych w ramach wątków zbioru* $u_b \in U_P$, *wykorzystujące wspólne zasoby* $R_P = \{r_c, ..., r_k\}$ *(tzn.* $(o_{m,j}, r_c) \in F_P$ *lub* $(r_c, o_{m,j}) \in F_P$, *...,* *lub* $(o_{n,q}, r_k) \in F_P$ *lub* $(r_k, o_{n,q}) \in F_P$*).*

Jeżeli istnieje taki dwuelementowy zbiór $O_P^* \subseteq O_P$, *który nie jest podzbiorem żadnej blokady* $m_l^b \in MU_P^b$*(tzn.* $\forall m_l^b \in MU_P^b : O_P^* \not\subseteq m_l^b$*) to zachodzi zjawisko szkodliwej rywalizacji między operacjami zbioru* O_P.

D. Giebas, R. Wojszczyk, „Multithreaded Application Model”, 19 międzynarodowa konferencja Distributed Computing and Artificial Intelligence, praca zaakceptowana

- Opracowanie modelu
- Twierdzenie o szkodliwej rywalizacji

Plan dalszych prac

- Twierdzenie o zakleszczeniu
- Twierdzenie o naruszeniu niepodzielności
- Twierdzenie o naruszeniu porządku
- Formalizacja transformacji kod → model
- Weryfikacja metody w praktyce

Spis treści

Publikacje

- D. Giebas, R. Wojszczyk, „Graphical representations of multithreaded applications”, Applied Computer Science, 14(2), Politechnika Lubelska, strony 20-37, Lublin 2018
- D. Giebas, R. Wojszczyk, „Zastosowanie wybranych reprezentacji graficznych do analizy aplikacji wielowątkowych”, Zeszyty Naukowe Wydziału Elektroniki i Informatyki, Wydawnictwo Uczelniane Politechniki Koszalińskiej, strony 5-25, Koszalin 2018
- D. Giebas, R. Wojszczyk, „Multithreaded Application Model”, 19 międzynarodowa konferencja Distributed Computing and Artificial Intelligence, praca zaakceptowana

Podsumowanie - Dalszy kierunek badań

- Transformacja odwrotna i automatyczne naprawianie kodu
- Zastosowanie modelu z innymi bibliotekami niż pthread
- Przystosowanie modelu do paradygmatu programowania obiektowego

mgr inż. Damian Giebas

Dziękuję za uwagę.