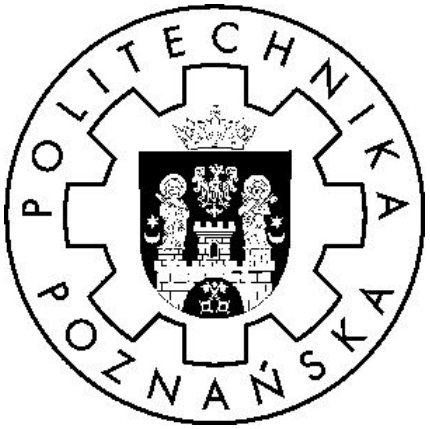

Zasady zaliczenia przedmiotu

Przetwarzanie Rozproszone

Wstęp

Ten dokument określa zasady zaliczenia przedmiotu *Przetwarzanie Rozproszone*, a w szczególności określa podstawowe wymogi dotyczące realizacji projektów zaliczeniowych oraz sprawozdań.

Terminy zaliczenia oraz zasady szczegółowe dostępne są na stronie prowadzących przedmiotu, których adresy są podawane na laboratoriach poświęconych wyborze problemów do rozwiązania.

Zaliczenie

Zakładając, że student nie zostanie wykluczony z zajęć z powodów regulaminowych (np. brak odpowiedniej ilości obecności), zaliczenie odbywa się na podstawie obrony projektów zaliczeniowych. Studenci i studentki dobierają się w dwuosobowe grupy, z których każda następnie otrzymuje problem do rozwiązania. Problemy związane są zwykle z dostępem do rozproszonej sekcji krytycznej. Następnie każda grupa przygotowuje i prezentuje do akceptacji algorytm. Algorytm jest poprawiany tak długo, aż zyska akceptację prowadzącego i dopiero wtedy grupa może przystąpić do implementacji algorytmu.

Obrona projektu polega na omówieniu algorytmu i jego implementacji, uruchomieniu programu ze zmienną liczbą parametrów, oraz przekazaniu wydrukowanego i podpisanego ręcznie sprawozdania.

Oceny wystawiane są indywidualnie, uwzględniając wkład członków grupy w rozwiązanie, jakość algorytmu, implementacji oraz sprawozdania. Zwykle członkowie grupy otrzymują taką samą ocenę, jednakże teoretycznie jest możliwe, że jeden student/studentka z pary otrzyma ocenę niedostateczną, a drugi/druga bardzo dobrą - zdarzyć się tak może w przypadku drastycznej asymetrii we wkładzie w ostateczne rozwiązanie.

Terminy

Dokładne terminy podawane są przez prowadzącego. Zwykle tematy problemów są wybierane w połowie semestru (na szóstych lub siódmych laboratoriach). Po dwóch tygodniach studenci proponują algorytmy. Na ostatnich zajęciach należy pokazać gotowe rozwiązania, uruchamiając je na co najmniej trzech maszynach. Oddanie projektu w terminie

późniejszym skutkować będzie obniżeniem oceny o pół stopnia za każde dwa tygodnie opóźnienia.

Prowadzący może zadać pytania każdej osobie z grupy dotyczące algorytmu lub kodu. Prowadzący może także polecić wprowadzenie modyfikacji kodu.

Sprawozdanie

Sprawozdanie musi zawierać opis problemu oraz algorytmu w postaci słownej oraz w postaci pseudokodu (najlepiej jako maszyna stanów). Algorytm powinien pokazywać możliwe stany procesów, wiadomości wysyłane w każdym stanie, reakcje na odbiór wiadomości danych typów w zależności od stanu procesów, oraz warunki przejścia między stanami. Sprawozdanie musi również zawierać opis złożoności czasowej oraz komunikacyjnej.

Opis podany w sprawozdaniu powinien być kompletny, szczegółowy i nie powinien wymagać dodatkowych wyjaśnień podczas obrony projektu. Należy przyjąć, że dowolna osoba powinna być w stanie zaimplementować algorytm na podstawie jego opisu w sprawozdaniu.

Założenia dotyczące środowiska

Środowisko jest w pełni asynchroniczne, kanały są niezawodne i FIFO, procesy nie ulegają awarii. W związku z tym nie można zakładać, że jeżeli proces odczeka kilka sekund, to coś się zmieni w stanach innych procesów: zegary procesów nie są zsynchronizowane i funkcjonują z różną prędkością. Procesu działają w wiecznej pętli.

Wymogi dotyczące rozwiązania

Algorytmy powinny być maksymalnie rozproszone. Nie powinny dopuszczać elementów centralnych, procesy powinny mieć równorzędne role. Niedopuszczalne jest rozwiązanie, w którym procesy zakładają globalne zamki na jakieś struktury, aktualizują je i potem przesyłają.

Procesy powinny wypisywać informacje o swoim stanie: o próbie dostępu do sekcji krytycznej (zasobu), o otrzymaniu dostępu, o zwolnieniu dostępu. Informacje powinny być poprzedzone identyfikatorem procesu i zegarem Lamporta.

Programy mogą być pisane w dowolnym języku programowania z użyciem MPI lub PVM, jednakże tylko dla kombinacji C+MPI gwarantowane jest wsparcie ze strony prowadzących. W pozostałych przypadkach należy samodzielnie zapewnić możliwość uruchomienia programu (co zwykle wymagać będzie kontaktu z administratorem laboratorium i doinstalowaniem odpowiednich pakietów).

Kod powinien być sparametryzowany, umożliwiając szybką zmianę liczby dostępnych zasobów (miejsc w sekcji krytycznej) i tak dalej, zgodnie z konkretnymi wymogami wybranego przez grupę tematu. Kod powinien być czytelny, z krótkimi funkcjami o nazwach ilustrujących ich przeznaczenie.

Algorytmy są optymalizowane pod względem minimalizowania formalnej złożoności czasowej i komunikacyjnej. Domyślnie należy przewagę dać minimalizacji formalnej złożoności czasowej (nie rzeczywistym czasom!), jednakże prowadzący może zlecić preferencję dla złożoności komunikacyjnej. W przypadkach szczególnych, jeżeli studenci

potrafią to właściwie uzasadnić, możliwe jest dopuszczenie algorytmów o większej złożoności formalnej, jeżeli posiadają one inne zalety.