

Systemy komputerowe

Lista zadań nr B

Na zajęcia 27 – 30 maja 2019

Należy przygotować się do zajęć czytając następujące rozdziały książek:

- Tanenbaum (wydanie czwarte): 2.2
- Stallings (wydanie dziewiąte): 4.1 – 4.3

UWAGA! W trakcie prezentacji rozwiązań należy zdefiniować i wyjaśnić pojęcia, które zostały oznaczone **wytłuszczoną** czcionką.

Zadanie 1. Czym różni się **przetwarzanie równoległe** (ang. *parallel*) od **przetwarzania współbieżnego** (ang. *concurrent*)? Czym charakteryzują się **procedury wielobieżne** (ang. *reentrant*)? Podaj przykład procedury (a) wielobieżnej, ale nie **wątkowo-bezpiecznej** (ang. *thread-safe*) (b) na odwrót. Kiedy w jedno-wątkowym procesie uniksowym może wystąpić współbieżność?

Wskazówka: Rozważ procedury obsługi sygnałów.

Zadanie 2. Rozważamy **wątki przestrzeni jądra** (ang. *kernel-level threads*). Czym różni się **przełączanie kontekstu** od **przełączania trybu pracy**? Gdzie jądro zachowuje kontekst wątku w trakcie przejścia z przestrzeni użytkownika do przestrzeni jądra? Czemu przełączanie między wątkami należącymi do różnych procesów jest bardziej kosztowne niż w obrębie tego samego procesu? Które z zasobów wymienionych w tabeli 2.4 (§2.1.6) należą do wątku, a które do procesu?

Zadanie 3. Wymień przewagi wątków przestrzeni jądra nad **wątkami przestrzeni użytkownika** (ang. *user-level threads*) zwanych dalej włóknami. Czemu biblioteka ULT musi kompensować brak wsparcia jądra z użyciem **opakowań funkcji** (ang. *wrapper*)? Jakie zdarzenia wymuszają przełączenie kontekstu między włóknami? Co dzieje się w przypadku kiedy włókno spowoduje błąd strony?

Zadanie 4. Opisz hybrydowy model wątków bazujący na **aktywacjach planisty** i pokaż, że może on łączyć zalety KLT i ULT. Posługując się artykułem [An Implementation of Scheduler Activations on the NetBSD Operating System](#)¹ wyjaśnij jakie **wezwania** (ang. *upcall*) dostanie planista przestrzeni użytkownika gdy zwiększymy lub zmniejszymy liczbę **wirtualnych procesorów**, wątek zostanie zablokowany lub odblokowany w jądrze, proces otrzyma sygnał.

Zadanie 5. Wyjaśnij jak przy pomocy **multipleksowania wejścia-wyjścia** (ang. *I/O multiplexing*) i **nieblokujących** wywołań systemowych zbudować jednowątkowy serwer TCP obsługujący współbieżnie wiele połączeń sieciowych. Zapoznaj się **notatkami**² do §6 książki „Unix Network programming, Volume 1”. Przeanalizuj kod **serwera**³ usługi echo wykorzystującego wywołanie **poll(2)**⁴. Opisz znaczenie flag zawartych w polach «events» i «revents» struktury pollfd.

Zadanie 6. Najpowszechniej implementowane wątki przestrzeni jądra wprowadzają do programów dodatkowy stopień złożoności. Co dziwnego może się wydarzyć w wielowątkowym procesie uniksowym gdy:

- wołamy funkcję `fork`, aby utworzyć podproces,
- użytkownik wciska kombinację «CTRL+C» i w rezultacie jądro wysyła «SIGINT» do procesu,

¹<http://web.mit.edu/nathanw/www/usenix/freenix-sa/freenix-sa.html>

²<https://notes.shichao.io/unp/ch6/>

³<https://github.com/shichao-an/unpv13e/blob/master/tcpcliserv/tcpservpoll101.c>

⁴<http://netbsd.gw.com/cgi-bin/man-cgi?poll>

- czytamy w wielu wątkach ze zwykłego pliku korzystając z jednego deskryptora pliku,
- modyfikujemy zmienną środowiskową funkcją `putenv(3)`⁵,
- wątki intensywnie korzystają z menadżera pamięci z użyciem procedury `malloc`.

Zadanie 7. Wątki nie są panaceum na problemy z wydajnością oprogramowania na **maszynach wieloprocessorowych ze współdzieloną pamięcią** (ang. *Shared Memory Processors*). Wymień warunki jakie musi spełniać architektura programu by stosowanie wątków było uzasadnione (§4.3)? Co ogranicza wydajność programów używających wątków? Jakie problemy z efektywnym użyciem wątków napotkali twórcy silnika gry *Half-Life 2*⁶ i jak je rozwiązali?

⁵<http://man7.org/linux/man-pages/man3/putenv.3.html>

⁶<https://arstechnica.com/gaming/2006/11/valve-multicore/>