prof. dr hab. inż. Tadeusz Czachórski
Instytut Informatyki, Wydział AEiI Politechniki Śląskiej

Gliwice, 20 marzec 2012 r.

Recenzja pracy doktorskiej

Autor rozprawy: mgr inż. Arkadiusz Chrobot

Tytuł rozprawy: ***Wirtualizacja rozproszonej pamięci operacyjnej multikomputera dla systemu Linux w oparciu o koncepcję SDDS***

Promotor rozprawy: prof. dr hab. inż. Krzysztof Sapiecha

Cel i zakres rozprawy.

Praca dotyczy ważnego i aktualnego problemu organizacji dostępu do pamięci w systemach wielokomputerowych, a więc systemów złożonych z oddzielnych komputerów, które nie mają wspólnej przestrzeni adresowej, a komunikacja między nimi odbywa się poprzez komunikaty przesyłane szybką siecią lokalną. Praca doskonali Skalowalne Rozproszone Struktury Danych – Scalable Distributed Data Structures (SDDS). Jest to aplikacja wirtualizująca pamięć RAM multikomputera na poziomie aplikacji użytkownika. Część węzłów pełni role klientów, a pozostałe węzły są serwerami przechowującymi rekordy danych w swoich pamięciach operacyjnych. Aplikacje widzą pamięć podzieloną na *wiaderka* i zapełnianą *rekordami*. Klient SDDS zna parametry pliku (obraz) potrzebne do ustalenia adresu serwera przechowującego określony rekord, brak jest katalogu centralnego dla adresacji danych, aktualizacja obrazu danych odbywa się za pomocą komunikatów korygujących; w przypadku, gdy klient posługuje się przedawnionym obrazem, serwer SDDS, do którego klient błędnie się zwrócił powinien skierować żądanie do właściwego serwera, a klientowi przesłać odpowiedni komunikat. Podstawową wadą istniejących implementacji SDDS jest to, że ich wykorzystanie wymaga zmian w kodzie aplikacji użytkownika.

Teza pracy stwierdza, że można tego uniknąć implementując część kliencką SDDS na poziomie systemu operacyjnego i że można uzyskać w ten sposób rozwiązanie skalowalne, zwiększające efektywność wykonania niektórych typów aplikacji. Praca przedstawia opis architektury proponowanego rozwiązania, opis jego implementacji, opis przeprowadzonych testów i analizę jego efektywności, wskazując aplikacje, dla których zaproponowane rozwiązanie jest korzystne.

Struktura pracy

Rozdział pierwszy omawia tezę pracy na tle opisu systemów wielokomputerowych, ich klasyfikacji i charakterystyki stosowanych architektur.

Rozdział drugi jest poświęcony dokładniejszemu opisowi wirtualizacji pamięci w systemach wielokomputerowych. Wychodząc od opisu pamięci wirtualnej w systemach jednoprocesorowych, Autor omawia pamięć wirtualną w tych systemach, pojęcia stronicowania na żądanie, segmentacji na żądanie, i stronicowanej segmentacji na żądanie. Wirtualizacja pamięci w systemach wielokomputerowych polega na agregacji pamięci lokalnych poszczególnych węzłów dla uzyskania puli pamięci dostępnej dla wszystkich systemów komputerowych w multikomputerze. Autor dokładnie i metodycznie omawia główne kierunki badań nad wirtualizcją pamięci rozproszonej multikomputerów: wsparcie dla pamięci operacyjnej (urządzenia wymiany, RAM dyski), sprzętowa, programowa i mieszana implementacja rozproszonej pamięci dzielonej, wsparcie dla usług zdalnych. Rozdział ten jest bardzo szczegółowym przeglądem rozwiązań, w oparciu o bogatą literaturę i dobrze świadczy o znajomości tych zagadnień przez Autora. Autor potrafił tez w podsumowanie przedstawić syntetyczne wnioski dotyczące spójności, wydajności odporności na błędy skalowalności i wykorzystania dostępnej pamięci operacyjnej w omawianych rozwiązaniach. W *Rozdziale trzecim* Autor powraca do tezy pracy, przedstawiając uzasadnienie jej podjęcia. Za SDDS przemawiają następujące argumenty: SDDS tworzą spójny obraz pamięci, którego obszar dostosowuje się do potrzeb aplikacji, nie posiadają centralnego katalogu, który utrudniałby skalowalność, czas pracy algorytmów adresowania nie jest zależny od wielkości pliku, a więc lokalizacje danych jest bardzo szybka. Przeciwko nim świadczy fakt, że trzeba je implementować w przestrzeni użytkownika. Stworzenie warstwy pośredniej oprogramowania częściowo usuwa te niedogodność, możliwe jest też lub udostępnienie aplikacjom SDDS jako usługi systemu operacyjnego. Trzeba jednak wykorzystać wywołania już istniejące - np. związane z obsługą plików. Można stworzyć

klienta SDDS, który byłby systemu operacyjnego aktywowanym przez wywołania aplikacji - może to być system plików lub sterownik urządzeń blokowych - Autor skłania się ku temu ostatniemu rozwiązaniu – przedstawieniu SDDS aplikacjom użytkowym jako urządzenia blokowego. Dodatkowym argumentem jest fakt, że takie urządzenia istnieją dla urządzeń pamięci masowej - pozwalają przedstawić urządzenie pamięci masowej jako tablicę, której każdy element (tzn. sektor) ma unikalny indeks w postaci liczby naturalnej w wielkość co najmniej 512 bajtów. Możliwe są operacje blokowe zapisu/odczytu wielu sąsiednich sektorów. Zbiór wszystkich wiaderek wchodzących w skład SDDS jest widziany jako rozproszona tablica haszujące DTH. Trzy podstawowe architektury SDDS przedstawiane w literaturze to LH* korzystająca z haszowania liniowego, RP* stosująca technikę podziałów zakresów, pozwalająca przechowywać rekordy w sposób uporządkowany oraz SD-Rtree używająca adresowania wieloindeksowego do lokalizacji danych przestrzennych. Autor wybiera pierwsze rozwiązanie i realizując tę architektur w systemie Linux – stąd nazwa SDDSfL.

Rozdział czwarty opisuje architekturę SDDSfL, rozpoczynając od opisu algorytmu haszowania liniowego, poprzez analizę możliwości przeniesienia SDDS na poziom systemu operacyjnego, po przedstawienie schematu, architektur klienta, serwera i koordynatora SDDSfL, jak również protokołów systemu. Poruszono tez sprawę odporności prototypu na awarie. *Rozdział piąty* opisuje prototypową implementację SDDSfL. Serwery SDDSfL są uruchamiane jako programy na osobnych węzłach multikomputera. Oprogramowanie klienckie zostało zrealizowane w postaci modułu jądra będącego sterownikiem urządzenia blokowego, tworzącym osobny watek dla obslugi transmisji między klientem a serwerem. Dedykowany mechanizm zajmuje sie retransmisją zagubionych pakietów. Informacje o trwałych błędach sterownika przekazywane są aplikacjom użytkowym za pomocą sygnałów. Wiaderka, którymi zarządzają serwery, są realizowane poprzez wykorzystanie tablic wskaźników, dynamiczna alokacje pamięci i blokowanie wymiany stron. Do adresowania bloków wewnątrz wiaderka użyto algorytmu adresowania otwartego z podwójnym haszowaniem. Trzy wątki kodu serwera realizują obslugę żądań klienta i operacje podziału wiaderka. Omówiono trudności realizacji: osadzenie klienta na poziomie jądra systemu operacyjnego powoduje, ze błąd w jego kodzie prowadzi do destabilizacji całego systemu, niełatwo wtedy zlokalizować sam błąd; na tym poziomi brak także bibliotek dostępnych aplikacjom, rozproszona struktura

SDDSfL także nie pomaga w lokalizacji błędów związanych z komunikacją.

Rozdział szósty przedstawia ocenę eksperymentalną – zawiera wyniki testów którym poddano powstałą implementację. Testowano SDDSfL z kilkoma rodzajami oprogramowania:

– transakcyjne bazy danych – wybrano PostgreSQL i program testowy `pgbench` badając liczbę transakcji w ciągu sekundy;

– procesy zorientowane na operacje wejścia-wyjścia – wybrano aplikację sortująca pliki rekordów zgodnie z algorytmem QuickSort, mierzono czas wykonania;

– procesy zorientowane na obliczenia były reprezentowane przez aplikację poszukująca wzorców tekstowych zgodnie z algorytmem Boyera-Moore'a, mierzono czas wykonania;

– podsystemy wymiany stron pamięci – program zapisujący zera w przydzielonym mu obszarze pamięci, wielkością mierzną jest czas zapisu.

Testy wykonano na multikomputerach zbudowanych na typowych komputerach klasy PC i sieci LAN oraz na systemie wielokomputerowym klasy MPP, wyposażonym w sieć lokalną Gigabit Ethernet oraz InfiniBand – standard otwartego, szeregowego, wielokanałowego interfejsu I/O o wysokiej przepustowości i niskim opóźnieniu. Wyniki testów zostały dokładnie przedstawione w postaci wykresów i omówione. W dyskusji wzięto pod uwagę parametry wykorzystywanego sprzętu. starano się określić skalowalność implementacji. Potwierdzono, ze zastosowane rozwiązanie tworzy efektywną metodę wirtualizacji pamięci, która może zastąpić rozproszone systemy plików lub urządzenia blokowe w zastosowaniach wymagających intensywnych operacji wejścia/wyjścia i swobodnego dostępu do danych.

Rozdział siódmy zawiera wnioski, w których m.in. przedstawiono zalety i wady SDDSfL: w przeciwieństwie do oryginalnych implementacji SDDS przedstawione rozwiązanie nie wymaga ingerencji w kod źródłowy aplikacji, prototyp wykazuje dobrą skalowalność, co jest ważne w przypadku, gdy trudno przewidzieć a priori, jaki będzie rozmiar zapisywanych informacji, ferowane czasy dostępu są rozsądne. Dodatek A omawia architektury SDDS odporne na błędy. Ten stosunkowo krótki tekst można było moim zdaniem umieścić w głównym tekście rozprawy. Bibliografia zawiera 117 pozycji, w tym jedną, której Doktorant jest współautorem, w materiałach *International Symposium on Parallel and Distributed Computing.* Przedstawiona lista pozycji jest wyczerpująca i dobrze świadczy o znajomości

przez Autora literatury przedmiotu.

Ocena rozprawy. Praca dotyczy ważnego i aktualnego problemu efektywnej organizacji dostępu aplikacji do pamięci w systemach wielokomputerowych. Zaproponowane rozwiązanie jest oryginalne, zostało jasno opisane i zaimplementowane, a jego efektywność została dobrze doświadczalnie przebadana. Autor wykazał się bardzo dobrą orientacją w istniejących rozwiązaniach i wyobraźnią, jak można je modyfikować. Część doświadczalna obejmuje dużą liczbę testów wykonanych przy użyciu aplikacji o zróżnicowanym charakterze i została dobrze metodycznie przeprowadzona, udokumentowana, a uzyskane wyniki szczegółowo omówione. Wyniki pokazują, że zaproponowane rozwiązanie charakteryzuje się porównywalną lub lepszą od tradycyjnych rozwiązań wydajnością, a jest przezroczyste dla pracy aplikacji, co jest duża zaletą organizacji dostępu do wspólnej pamięci. Wyniki potwierdzają więc jednoznacznie tezę pracy. Być może dałoby się uzupełnić przeprowadzone testy bardziej ogólną analizą efektywności pracy zaproponowanej architektury, ale jest to całkiem oddzielne wyzwanie.

Praca jest napisana jasno, choć zdarzają się drobne błędy edytorskie. Układ pracy jest logiczny. Całość świadczy o dobrym przygotowaniu Doktoranta do pracy naukowej. Rezultaty zasługują na dalsze publikacje.

Wniosek końcowy.

Podsumowując, uważam, że rozprawa doktorska mgr inż. Arkadiusza Chrobota spełnia warunki stawiane rozprawom doktorskim przez ustawę o stopniu i tytułach naukowych. Autor wykazał się dużą znajomością zagadnień związanych z działaniem systemów rozproszonych, potrafił zaproponować, zastosować i szczegółowo zbadać oryginalną metodę zwiększenia ich efektywności. Wnioskuję o przyjęcie tej pracy jako rozprawy doktorskiej i dopuszczenie jej do publicznej obrony.