STUDIA INFORMATICA 2003
Volume 24 Number 3 (55)

Dariusz Rafał AUGUSTYN, Łukasz WYCIŚLIK
Politechnika Śląska, Instytut Informatyki

ZASTOSOWANIE TECHNOLOGII ROZPROSZONEGO PRZETWARZANIA DCOM FIRMY MICROSOFT CORP. W ZRÓWNOLEGLONEJ REALIZACJI ZADANIA OPTYMALIZACJI PARAMETRYCZNEJ

Streszczenie. W artykule przedstawiono system rozproszonej realizacji zadania optymalizacji parametrycznej pracujący w architekturze nadzorca-wykonawcy. Zastosowanie technologii DCOM firmy Microsoft w omawianym systemie pozwoliło na komunikację programu nadzorcy i komponentów-wykonawców, uruchomionych w różnych węzłach lokalnej sieci komputerowej. Artykuł pokazuje efektywnościowe korzyści przy zastosowaniu omawianego systemu w rozwiązywaniu wielowymiarowej optymalizacji parametrycznej.

Słowa kluczowe: wielowymiarowa optymalizacja parametryczna, przetwarzanie rozproszone, technologia DCOM.

THE DISTRIBUTED PROCESSING TECHNOLOGY DCOM OF MICROSOFT CORPORATION IN PARALLEL EXECUTION OF A PARAMETRIC OPTIMIZATION TASK

Summary. This article presents distributed system of execution a parametric optimization task, based on master-workers architecture. The Microsoft DCOM technology was used for communication between these elements of the system which all should be run on different nodes of LAN. The efficient advantages of the discussed system in solving a multidimensional optimization problem were shown.

Keywords: multidimensional parametric optimization, distributed processing, DCOM technology.

1. Wstęp

Zadanie statycznej optymalizacji parametrycznej jest problemem występującym w modelowaniu wielu zjawisk fizycznych, biologicznych czy ekonomicznych. Zadanie to polega na znalezieniu minimum pewnej funkcji, zwanej wskaźnikiem jakości, której wartość zależy od pewnego zbioru parametrów [1]. Zależność ta jednak nie jest jawna i do wyznaczenia pojedynczej wartości wskaźnika jakości niezbędne jest wykonanie czasochłonnego eksperymentu dla zadanych wartości parametrów. Zadanie minimalizacji wskaźnika jakości jest rozwiązywane na ogół w sposób numeryczny i pociąga za sobą wielokrotne (aż do osiągnięcia zakładanej dokładności) wykonywanie eksperymentów (np. całkowania równań stanu opisujących model). Rezultatem działania algorytmu optymalizacji parametrycznej jest taki wektor wartości parametrów, dla których wielowymiarowa funkcja wskaźnika jakości przyjmuje wartość najmniejszą.

Niektóre algorytmy zadania optymalizacji realizujące minimalizację wielowymiarowej funkcji jakości pozwalają na zrównoleglenie działań, a przez to efektywniejszą realizację w rozproszonym systemie przetwarzania. Przykładem takiego prostego algorytmu może być takie wyszukiwanie minimum funkcji wskaźnika, gdzie w kroku roboczym przyjmuje się ustalone wartości parametrów z wyjątkiem jednego, dla którego przeprowadza się minimalizację już jednowymiarowej funkcji. Każdy taki krok roboczy, w którym następuje uzmiennienie wybranego parametru przy ustaleniu pozostałych, jest niezależny od "ortogonalnych" kroków roboczych (tzn. kroków dla innych uzmiennianych parametrów). Ta niezależność pozwala na w pełni równoległą realizację kroków roboczych. Zgromadzone wszystkie wyniki wykonania kroków roboczych (tzn. wartości poprzednio uzmiennianego parametru, dla którego wystąpiło minimum jednowymiarowego wskaźnika) pozwalają na wykonanie kroku zasadniczego metody, czyli zmiany wszystkich parametrów.

Równoległa realizacja zadania wg algorytmu opisanego powyżej może być realizowana w klasycznej rozproszonej architekturze przetwarzania, w której występują program nadzorczy (ang. *master*) i programy wykonawców (ang. *workers*). Program nadzorczy realizuje iteracyjnie kroki zasadnicze metody (rozsyła wektor wartości parametrów do wykonawców wskazując parametr uzmienniany, odbiera wartości minimalne poprzednio uzmiennianych parametrów, wyznacza nowy wektor parametrów), zatrzymuje proces optymalizacji po uzyskaniu zakładanej dokładności znalezionego wektora. Programy wykonawców realizują kroki robocze. Najbardziej efektywna realizacja algorytmu może mieć miejsce dla konfiguracji, w której każdy z programów (nadzorca i wykonawcy) jest uruchomiony na różnych komputerach, tzn. w różnych węzłach sieci komputerowej. Opisywany przykładowy algorytm optymalizacji może być efektywnie realizowany w środowisku sieci

komputerowych nie tylko ze względu na niezależność zadań wykonawców, ale i niewielkie, w sensie zajętości, rozmiary przesyłanych danych.

Artykuł prezentuje dedykowany system, realizujący rozproszoną wersję kilku wybranych algorytmów optymalizacji parametrycznej, bazujący na technologii COM/DCOM firmy Microsoft Corp. (ang. Distributed Component Object Model [2, 3, 7]), charakterystycznej dla systemów MS Windows 9x/2000/XP. Omawiany system składa się z komponentów COM/DCOM, instalowanych na różnych komputerach sieci, realizujących zadania wykonawców. Komponenty DCOM wykonawców udostępniają metody pozwalające na: inicjalizację nadzorcy (m.in. przesłanie opisu modelu w postaci równań stanu, wyrażeń na wskaźnik jakości albo jego pochodne), uruchomienie kroku roboczego (algorytm kroku roboczego jest uzależniony od metody optymalizacji), odesłanie wyniku roboczego. Nadzorca, w postaci wielowątkowego programu, pracuje jako klient usług DCOM.

2. Określenie zadania statycznej optymalizacji parametrycznej

Poniżej przedstawione zostało ogólnie sformułowane zadanie statycznej optymalizacji parametrycznej [1]. Dany jest układ dynamiczny, opisany równaniem stanu:

$$\frac{d\underline{x}}{dt} = f(\underline{x}, t, \underline{\alpha}), \tag{1}$$

gdzie:

$\underline{x}^T = [x_1, x_2, ..., x_N]$ - wektor stanu,

$\underline{\alpha}^T = [\alpha_1, \alpha_2, ..., \alpha_m]$ - wektor parametrów.

Układ ten obserwowany jest w przedziale czasu $t \in \langle t_0, t_{\max} \rangle$. Warunki początkowe wektora stanu mogą być określone lub mogą być zależne od składowych wektora $\underline{\alpha}$, co zapisujemy równaniem:

$$\underline{x}(t_0) = \varphi(\underline{\alpha}). \tag{2}$$

Przyjmujemy, że składowe wektora $\underline{\alpha}$ mogą oznaczać parametry modelu, jeżeli występują jawnie w równaniu stanu lub mogą definiować wartości początkowe składowych wektora stanu, gdy występują jawnie w równaniach (2). Niezależnie od znaczenia składowych wektora $\underline{\alpha}$ przyjmuje się, że nie zmieniają one wartości w przedziale obserwacji $\langle t_0, t_{\max} \rangle$.

Rozwiązanie równania stanu (1) jest funkcją zmiennej niezależnej t oraz funkcją składowych wektora parametrów $\underline{\alpha}$, co podkreślamy zapisując rozwiązanie w postaci:

$$\underline{x} = \underline{x}(t, \alpha). \tag{3}$$

W zadaniach modelowania, których celem jest dobór parametrów modelu, wprowadza się zazwyczaj pewną charakterystykę – funkcjonał, który przypisuje każdemu rozwiązaniu (3) liczbę, określającą przydatność tego rozwiązania. Charakterystyki tego typu nazywane są wskaźnikami jakości i oznaczane zapisem:

$$Q(\underline{\alpha}) = \phi(\underline{x}(t, \underline{\alpha})), \tag{4}$$

gdzie funkcjonał ϕ jest określony analitycznie (jawnie).

Funkcja $Q(\underline{\alpha})$ oznacza tu zależność wskaźnika jakości od wektora parametrów $\underline{\alpha}$. Funkcja ta nie jest znana analitycznie, gdyż nie jest znana zazwyczaj analityczna postać rozwiązania (3). Wyznaczenie wartości wskaźnika jakości Q dla zadanego wektora $\underline{\alpha}$ wymaga każdorazowo rozwiązania układu równań stanu (1).

Zadanie optymalizacji parametrycznej sprowadza się do doboru wartości składowych wektora $\underline{\alpha}$ ze zbioru dopuszczalnych wartości A, ($\underline{\alpha} \in A$), tak by wskaźnik jakości (4) przyjmował minimalną wartość.

Nieco inaczej sformułowane zadanie optymalizacji parametrycznej polega na znalezieniu takiej wartości wektora $\underline{\alpha}'$ ze zbioru dopuszczalnych wartości A ($\underline{\alpha}' \in A$), by dla każdego $\underline{\alpha} \in A$ spełniona była nierówność $Q(\underline{\alpha}') \leq Q(\underline{\alpha})$, co zapisujemy:

$$\min_{\underline{\alpha} \in A} Q(\underline{\alpha}) = Q(\underline{\alpha}'). \tag{5}$$

Omawiane w artykule algorytmy optymalizacji są algorytmami numerycznymi. Numeryczne rozwiązanie zadania optymalizacji polega na znalezieniu w sposób rekurencyjny, w kolejnych krokach, takiego ciągu wartości parametrów $\underline{\alpha}_1, \underline{\alpha}_2, \ldots, \underline{\alpha}_n, \ldots$, aby:

$$\lim_{n \to \infty} \underline{\alpha}_n = \underline{\alpha}'. \tag{6}$$

3. Charakterystyka zastosowanych algorytmów optymalizacji

W omawianym systemie zaimplementowane algorytmy bazują na następujących metodach optymalizacji [1]:

a) Zmodyfikowana metoda Gaussa-Seidela (G-S)

Metoda znajdywania minimum funkcji w zadanym kierunku poprzez kolejne uzmiennianie jednego z parametrów optymalizacji; pojedynczy krok roboczy, tzn. szukanie minimum funkcji jednej zmiennej odbywa się za pomocą algorytmu skończonego przyrostu (począwszy od zadanego punktu startowego) lub metodą złotego podziału (dla zadanego przedziału startowego); pojedynczy krok zasadniczy metody pozwala na znalezienie kolejnego wektora $\underline{\alpha}_{k+1}$ poprzez uwzględnienie zmian we wszystkich jego składowych

z poprzedniej iteracji $\underline{\alpha}_k$ na podstawie wartości znalezionych w krokach roboczych; zakończenie algorytmu może nastąpić, jeśli odległość (np. w metryce euklidesowej) pomiędzy kolejno uzyskanymi przybliżeniami $\underline{\alpha}_k$ i $\underline{\alpha}_{k+1}$ jest poniżej zadanej dokładności albo zmiana wartości wskaźnika jakości jest w kolejnych krokach poniżej zadanej dokładności.

b) Metoda simpleksu - Neldera-Meada (N-M)

Metoda, w której następuje "zmniejszanie" m-wymiarowego wielościanu o m+1 wierzchołkach w przestrzeni R^m parametrów optymalizacji; wielościan ten w kolejnych krokach "przesuwa" się w kierunku szukanego punktu minimum (szukanego wektora $\underline{\alpha}^*$) lub zmniejsza się, jeśli zawiera ten punkt; przesunięcia o wektor współrzędnych poszczególnych wierzchołków wielościanu odbywają się w oparciu o wyznaczone wartości wskaźnika jakości liczone w wierzchołkach (operacje: odbicia względem "środka ciężkości", ekspansji, kontrakcji i redukcji [1]); zakończenie algorytmu może mieć miejsce, jeśli wszystkie odległości pomiędzy każdymi dwoma wierzchołkami są poniżej zakładanej dokładności.

c) Metoda gradientowa (w wersji z wrażliwościami)

Metoda poruszania w kierunku największego spadku; kierunek największego spadku wyznaczony jest jako $-\nabla Q(\underline{\alpha}_k)$ - minus gradient z funkcji wskaźnika jakości (wektor pochodnych cząstkowych wskaźnika po parametrach optymalizacji); gradient może być wyznaczany metodą wrażliwości, które są pochodnymi zmiennych stanu po parametrach optymalizacji; wrażliwości pozwalają na wyznaczenie składowych gradientu w jednym eksperymencie (jednokrotne całkowanie równań stanu), ale rozszerzają opis modelu o kolejne równania stanu, tzn. o m x r równań, gdzie r to liczba zmiennych stanu, a m to liczba parametrów optymalizacji; kolejne przybliżenia szukanego $\underline{\alpha}^*$ liczone są jako:

$$\underline{\alpha}_{k+1} = \underline{\alpha}_k - b\nabla Q(\underline{\alpha}_k), \tag{7}$$

gdzie zmiana wartości skalara b pozwala na kontrolę szybkości zbieżności ciągu $\underline{\alpha}_n$.

4. Moduł wykonawcy

Moduł wykonawcy, w postaci komponentu DCOM, implementuje interfejsy poprzez metody realizujące podstawowe czynności, tj.: inicjalizację (odebranie równań stanu, równań wyjścia, translacji wyrażeń w równaniach stanu na ONP, odebranie wartości początkowych zmiennych stanu, czasu rozpoczęcia lub zakończenia eksperymentu, wartości parametrów, wybranie typu metody całkowania i inne, w zależności od algorytmu optymalizacji, np. dokładność obliczeń), sterowanie (rozpoczęcie wykonania, zaprzestanie pracy przez wykonawcę), wysłanie wyników (odesłanie wybranych zmiennych stanu, wyjścia, czy parametrów).

Komponenty DCOM wykonane zostały w środowisku VC++ 6.0 z wykorzystaniem biblioteki MFC. Obiektowa architektura pozwala na łatwe rozszerzanie funkcjonalności poprzez dodefiniowywanie metod, realizujących roboczy krok algorytmu wykonawcy (dla pewnych algorytmów optymalizacyjnych, jak np. simpleks, metoda odpowiedzialna jest jedynie za wyznaczenie wartości wskaźnika jakości, dla innych, np. bazujących na algorytmie G-S, metoda realizuje bardziej złożone przetwarzanie - wyszukiwanie minimum w zadanym kierunku, z zadaną dokładnością). Obiektowa architektura pozwala również na łatwe dołączanie kodu dodatkowych metod całkowania (oprócz zaimplementowanych już metod Eulera, Rungego-Kutty) oraz dodefiniowywania jednoargumentowych funkcji, które mogą wystąpić w wyrażeniach po prawych stronach równań stanu.

5. Moduł nadzorcy

Program nadzorcy obejmuje funkcjonalnie następujące części:

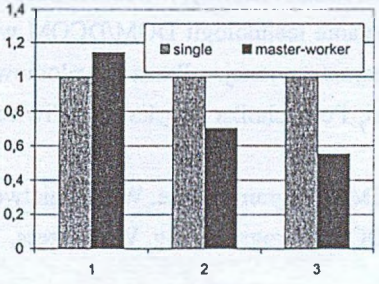
- moduł interfejsowy
  (pozwalający na odczytanie definicji zadania optymalizacji poprzez interfejs użytkownika (ekran i klawiatura) lub z pliku wsadowego (format INI lub XML) albo poprzez metody automatyzacji OLE),
- moduł realizacji zadania optymalizacji oraz kontroli i sterowania wykonawcami
  (sprowadzający się jedynie niemal do sterowania wykonawcami w metodzie bazującej na algorytmie G-S czy metodzie gradientowej albo realizujący zasadniczą część algorytmu optymalizacji, jak w przypadku algorytmu N-M).

Zastosowanie wielowątkowej architektury programu nadzorcy umożliwiło nie blokujące wywołania metod DCOM-owych obiektów wykonawczych (dla każdego wykonawcy powoływany jest wątek "komunikacyjny" po stronie procesu nadzorcy).

Program nadzorczy nie realizuje w obecnej implementacji [5] algorytmów równoważenia obciążeń. Nadzorca pobiera jednak wartości pewnych współczynników skojarzone z adresami sieciowymi komputerów, na których zainstalowane są komponenty DCOM. Wartości te (przy założeniu takiej interpretacji) mogą w sposób statyczny charakteryzować szybkość maszyny, na której potencjalny wykonawca pracuje. Program nadzorcy mierzy też czas wykonania kroku roboczego przez konkretnego wykonawcę. Wartości współczynników szybkości maszyny czy czasów realizacji zadań wykonawców pozwolą w przyszłości na rozszerzenie funkcji omawianego systemu o jakiś prosty wariant równoważenia obciążeń po stronie programu nadzorczego.

6. Wyniki testów

Pierwsze wyniki testów pokazały, że spodziewane przyspieszenie przetwarzania może mieć miejsce. Dla metody G-S (najbardziej obciążającej wykonawców) uśrednione wyniki testów pokazano na rys. 1. Przedstawiono tam czas wykonania zadania optymalizacji w systemie nadzorca-wykonawcy (*master-worker*) w funkcji liczby wykonawców (równej liczbie parametrów optymalizacji), wyskalowany czasem wykonania "klasycznego" programu o działaniu sekwencyjnym (*single*). Dla optymalizacji parametrycznej jednowymiarowej rozwiązanie klasyczne, sekwencyjne jest lepsze ze względu na czasowe narzuty rozwiązania nadzorca-wykonawca w komunikacji z jednym, lokalnym, wykonawczym obiektem COM. Dla optymalizacji ponadjednowymiarowej (2 lub 3 procesy wykonawców; wszystkie procesy na różnych komputerach) rozwiązanie nadzorca-wykonawcy jest lepsze.



Rys. 1. Względny czas wykonania zadania optymalizacji zmodyfikowaną metodą G-S w funkcji liczby wykonawców

Fig. 1. The relative time of execution optimization task of modified G-S method in dependency on number of workers

7. Zakończenie

System jest na etapie rozbudowy programu nadzorczego o funkcje serwera automatyzacji OLE (i tym samym również komponentu COM) [2, 3, 6]. Ze względu na otwartość takiego rozwiązania możliwe będzie korzystanie z usług serwera z poziomu większości aplikacji dla Windows (tych, które potrafią korzystać z interfejsu IDispatch).

Kolejnym etapem rozbudowy systemu będzie wykorzystanie nowej technologii Web Services w następnych edycjach systemu rozproszonej realizacji optymalizacji. Zastosowanie pakietu narzędziowego typu MS Soap Toolkit [4] pozwoli na automatyzację procesu opakowania obiektu COM nadzorcy w odpowiednią usługę WebServices (generacja opisowych plików WSDL, WSML). Umożliwi to uruchamianie zadań optymalizacji za pośrednictwem XML-owych komunikatów protokołu SOAP. Tym samym pozwoli to na

wykorzystanie omawianego systemu nie tylko użytkowników sieci lokalnych, ale również przez oddalonych użytkowników z węzłów sieci Internet.

LITERATURA

1. Seidel. J, Badach. A, Molisz W.: Metody rozwiązywania zadań optymalizacji. WNT, Warszawa 1980.
2. Eddon G., Eddon H.: Inside Distributed COM. Microsoft Press, Redmond, Washington, 1998.
3. Chappell D.: Understanding ActiveX and OLE. Microsoft Press, Redmond, Washington. 1996.
4. Dokumentacja techniczna: MSDN Library. Microsoft Corporation, 2002.
5. Ligudziński K.: Zastosowanie technologii DOM/DCOM w rozproszonej realizacji zadania optymalizacji parametrycznej. Praca dyplomowa, Wydział Automatyki, Elektroniki i Informatyki, Politechnika Śląska w Gliwicach, kierunek Informatyka, 2002.
6. Eddon G., Eddon H.: COM+ Programowanie. Wydawnictwo RM, Warszawa 2001.
7. GrimesR.: Proffesional DCOM Programming. Wrox Press, 1997.

Recenzent: Dr inż. Arkadiusz Sochan

Wpłynęło do Redakcji 29 kwietnia 2003 r.

Abstract

This paper presents a system of distributed execution of parametric optimization tasks. The system bases on master-workers architecture using in communication the Microsoft's COM/DCOM technology. Workers are implemented as DCOM components. A load of worker's tasks depends on optimization method but in all cases of used method workers must integrate differential state equations, which describe the modeled continuous system.

The article shows that in multidimensional optimization problem the discussed distributed system is more efficient than a classical sequential program.

The future implementation of master program as an OLE automation server or Web Services was announced, which lets a better integration of the discussed system with third party applications and lets remote Internet users to use the system.

Adresy

Dariusz Rafał AUGUSTYN: Politechnika Śląska, Instytut Informatyki, ul. Akademicka 16, 44-101 Gliwice, Polska, Rafal.Augustyn@iinf.polsl.gliwice.pl

Łukasz WYCIŚLIK: Politechnika Śląska, Instytut Informatyki, ul. Akademicka 16, 44-101 Gliwice, Polska, Lukasz.Wycislik@iinf.polsl.gliwice.pl.