バックトラックに基づく負荷分散の広域分散環境における評価

河野卓矢[†]　八杉昌宏[††]　平石拓[†††]
馬谷誠二[††]　湯淺太一[††]

1. はじめに

マルチコアプロセッサ等を含む並列計算環境が一般的になるに伴い，並列計算向け高生産性言語はより重要になっている．我々は，そのような環境において粒度が大きくバランスのとれた負荷分散を可能にするフレームワークTascell[2]を提案している．これは，負荷分散の要請時に図1のように一時的にバックトラックを行うことにより，並列化のコストを（例えば論理スレッドを利用するCilk[1]と比較して）本質的に極小化する遅延分割型負荷分散を実現するものである．

Tascellフレームワークは，Tascell言語コンパイラとTacsellサーバからなる．共有メモリ環境において，Tascellプロセスは一つ以上のワーカを持っており，アイドルなワーカは負荷のあるワーカにタスクを要求する．要求を受けたワーカはタスクを分割して新しいタスクを生成し，要求を出したワーカにそれを送る．タスクの要求をしたワーカはその新しいタスクを受けとり実行を開始する．Tacsellサーバに，複数のTascellプロセスを接続することで，計算ノードを跨いでタスクをやりとりすることができ，分散メモリ環境における並列計算も実現できる．実際，クラスタ環境において高い性能を得られることを既に確認している．

本研究では，多拠点のクラスタを相互に接続した計算環境InTrigger[3]にTascellを適用し，評価を行った．またそれに付随して，可搬性を高めるため，従来商用Lispで実装されていたTascellサーバのJavaへの移植も行った．

2. 広域分散環境への対応

Tascellを複数拠点のクラスタで連携させて実行するためには，まず，各拠点の代表ノードでTascellサーバを立ち上げ，それらをtreeを構成するように相互に接続する．次に各拠点の各計算ノードで計算プロセスを立ち上げ，それらを自分の拠点のサーバに接続する．

図2にInTrigger環境で2拠点を接続した場合を示す．chiba100のサーバは自分の計算ノードからタスク要求を受け付け，拠点内には分割できるタスクは残っていないと判断すると親であるimade000のサーバにタスク要求を転送する．imade000のサーバはこの要求を（計算ノードからのタスク要求の場合と同様に）自分の拠点内の計算ノードのいずれかに転送する．

拠点間の通信をサーバ間に限定することで，拠点間の通信回数を抑えることができるほか，全計算ノードがグローバルIPアドレスを持たないような環境にも対応できる．

3. 性能評価

InTriggerのうち，chiba，imadeの2拠点を使用して性能評価を行った．接続構成は図2の通りである．各拠点では最大16ノードを使用した．また，各ノードでは最大2ワーカで共有メモリ並列計算を行った．評価に使用したプログラムは，再帰版Fibonacci（fib(n)），n女王問題の全解探索（nq(n)），nピースPentominoパズルの全解探索（pen(n)）である．

表1に評価環境，表2に実行時間の測定結果を示す．

単一クラスタでの結果と，2クラスタでの結果を合計使用ノード数が同じもので比較すると，ほぼ同等の性能が得られており，サーバを拠点間で接続することによる通信レイテンシ等の影響はあまり見られなかった．特に，2クラスタでの性能が1クラスタの場合を上回っている場合があるが，これは，1サーバに接続されるノード数が増えることによるサーバの負荷の増大の影響が，拠点間通信の影響を上回ったためだと考えられる．

表1　評価環境

CPU	chiba, imade ともに Core2 Duo E6400 2.13GHz
OS	Linux 2.6.18
コンパイラ	gcc 4.4.2 (X86-64)　最適化オプション-O2
ワーカ	pthread_create で生成 (PTHREAD_SCOPE_SYSTEM スケジューリング)
サーバ	Java1.6.0_07
ネットワーク	Gigabit Ethernet (クラスタ内)，Sinet (クラスタ間)， サーバ・ノード間は TCP/IP で接続

† 京都大学工学部情報学科
†† 京都大学情報学研究科
††† 京都大学学術情報メディアセンター

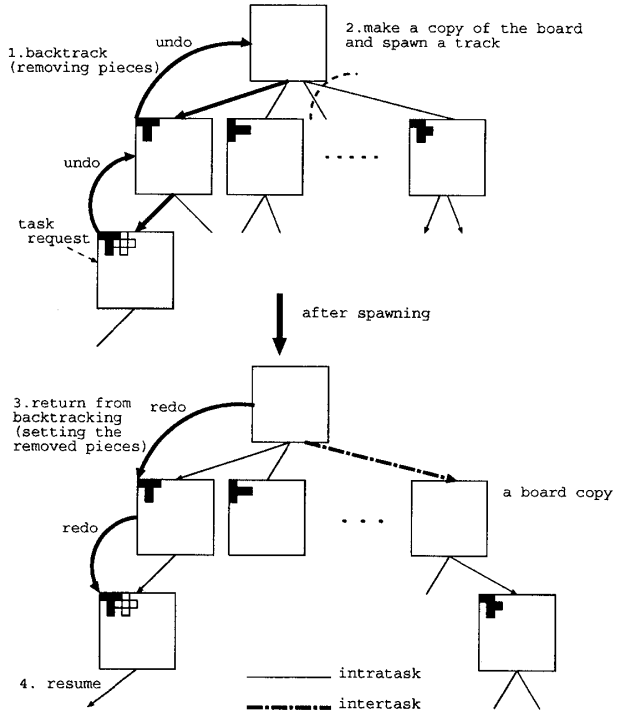

図 1 Pentomino 全解探索におけるタスク遅延生成．Tascell ワーカがタスク要求を二つ目のピースを置く際に検出すると，(2) 最古のタスク生成可能時点まで undo（ピース除去）しつつバックトラックし，(2) 反復の半分を新たなタスクとして生成し，(この時，一時的に過去の状態に戻った盤面をコピー)，(3) redo（ピースを再設置）しつつバックトラックから復帰．(4) 自らの計算を再開する．

ただし，2 クラスタでの計算においては，ノード数を増やすにつれて，特に pen(14) で性能が不安定になる傾向も見られた．例えば $T_1(4+4)$ の場合の計算が 40.0 秒かかることもあった．これは，偶発的に多くの小さなタスクの通信が拠点間で行われたことが原因であると考えられる．

4. おわりに

本研究では，動的負荷分散フレームワーク Tascell を広域分散環境に適用し，その評価を行った．今後は，通信回数などの詳細な評価や，より大規模な並列計算における評価を行い，広域環境に適した負荷分散戦略などの検討を行う予定である．

謝辞 本研究は文部科学省科学研究費補助金特定領域研究「情報爆発時代に向けた新しい IT 基盤技術の研究」の助成を得て行われた．

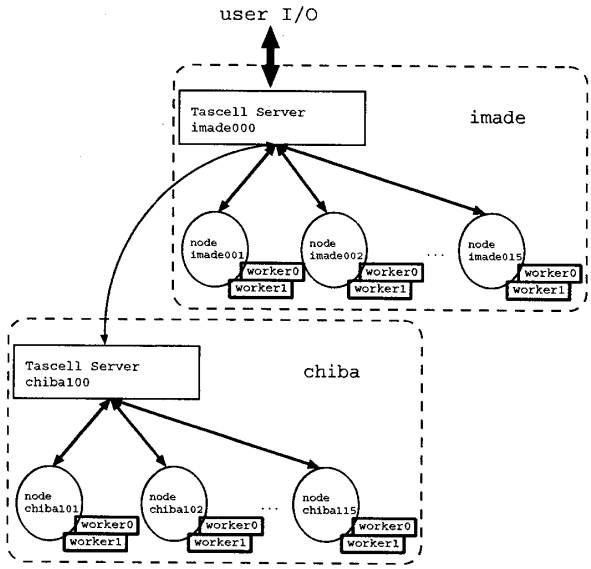

図 2 Tascell を広域分散環境で実行するための接続構成

参考文献

1) Frigo, M., Leiserson, C. E. and Randall, K. H.: The Implementation of the Cilk-5 Multithreaded Language, *ACM SIGPLAN Notices (PLDI '98)*, Vol.33, No.5, pp.212–223 (1998).
2) Hiraishi, T., Yasugi, M., Umatani, S. and Yuasa, T.: Backtracking-based Load Balancing, *Proceedings of PPoPP 2009* (2009).
3) 斎藤秀雄, 鴨志田良和, 澤井省吾, 弘中健, 高橋慧, 関谷岳史, 頓楠, 柴田剛志, 横山大作, 田浦健次朗: InTrigger：柔軟な構成変化を考慮した多拠点に渡る分散計算機環境, 情報処理学会研究報告-ハイパフォーマンスコンピューティング（HPC），Vol.2007, No.80, pp. 237–242 (2007).

表 2 実行時間の測定結果（秒）．「C」は C 言語による逐次プログラム．「$T_m(n)$」は単一クラスタで n ノード，「$T_m(n+n)$」は 2 拠点で各 n ノードを使用した Tascell（m は各ノード内のワーカ数）．括弧内の数は C に対する速度向上．

単一クラスタ（**chiba** のみ）
各ノード内 1 ワーカ

	C	$T_1(1)$	$T_1(2)$	$T_1(4)$	$T_1(8)$	$T_1(16)$
fib(44)	8.68	16.7(0.520)	8.43(1.03)	5.90(1.47)	3.18(2.73)	2.56(3.39)
nq(16)	75.5	86.7(0.871)	43.6(1.73)	28.3(2.67)	14.8(5.10)	9.55(7.91)
pen(14)	81.0	103(0.788)	54.4(1.49)	37.9(2.14)	20.8(3.89)	14.9(5.42)

各ノード内 2 ワーカ

	C	$T_2(1)$	$T_2(2)$	$T_2(4)$	$T_2(8)$	$T_2(16)$
fib(44)	—	9.76(0.888)	4.60(1.89)	3.35(2.59)	2.70(3.21)	2.94(2.95)
nq(16)	—	42.8(1.76)	22.1(3.42)	15.9(4.76)	10.6(7.13)	7.42(10.2)
pen(14)	—	51.3(1.58)	33.6(2.41)	18.5(4.38)	9.83(7.64)	8.30(10.9)

2 クラスタ（**chiba+imade**）
各ノード内 1 ワーカ

	$T_1(1+1)$	$T_1(2+2)$	$T_1(4+4)$	$T_1(8+8)$	$T_1(16+16)$
fib(44)	8.58(1.01)	5.23(1.66)	4.02(2.16)	2.23(3.89)	1.79(4.84)
nq(16)	43.7(1.73)	27.6(2.74)	18.4(4.10)	11.0(6.87)	6.73(11.2)
pen(14)	55.1(1.47)	36.5(2.22)	18.4(4.39)	14.4(5.61)	11.7(6.95)

各ノード内 2 ワーカ数

	$T_2(1+1)$	$T_2(2+2)$	$T_2(4+4)$	$T_2(8+8)$	$T_2(16+16)$
fib(44)	4.92(1.76)	2.89(3.00)	1.95(4.46)	1.84(4.73)	1.71(5.07)
nq(16)	22.3(3.39)	12.4(6.08)	8.11(9.31)	5.94(12.7)	5.03(15.0)
pen(14)	33.6(2.41)	16.9(4.80)	10.8(7.47)	10.6(7.62)	8.83(9.17)