Vol.2009-HPC-121 No.35
2009/8/6

性能を保証する計算・ネットワーク資源のコアロケーション手法の評価

竹房　あつ子[†1]　中田　秀基[†1]
工藤　知宏[†1]　田中　良夫[†1]

利用者の要求する性能を保証しつつ計算機・ネットワークなどの資源を組合わせて提供する場合，それらの資源のスケジューリングが重要な課題となる．我々は既発表研究において，このようなスケジューリングを最適化問題としてモデル化し，必要な計算・ネットワーク資源を同時に確保するためのオンラインコアロケーション手法を提案した．一方，最適化問題として解くと，変数の数に依存して指数的に求解時間が長くなるため，実用化に向けた課題が残る．本研究では，(1) 制約条件を追加することにより既発表研究の手法を改良し，求解時間の短縮を図るとともに，(2) 制約条件とソルバの違いによるに求解時間を比較し，実用化に向けて議論する．また，既発表研究に対して追加の実験を行い，(3) 重み付けすることにより管理者の要求を反映した資源の割り当てが可能であり，我々の手法が機能面でも有効であることを示す．

Study of Global Co-allocation for Performance-guaranteed Computing and Network Resources

Atsuko Takefusa,[†1] Hidemoto Nakada,[†1]
Tomohiro Kudoh[†1] and Yoshio Tanaka [†1]

Resource scheduling is the key technology to provide computing and network resources to the users, while meeting the QoS with the users. In our previous work we modeled the resource allocation problem into a combinatorial optimization problem and proposed an on-line co-allocation method to co-allocate required computing and network resources. While the proposed method worked fine, it was still not ready for production usage, since the resolution time became too long, depending on the number of variables. In this paper, toward production usage, we 1) improved the method by adding constraint to reduce resolution time, 2) evaluated constraints and solvers from the aspect of the resolution time, and conducted additional experiments and 3) proved that resource administrator's preference can be reflected by our method.

1. はじめに

利用者の要求する性能を保証する計算機，ネットワーク，ストレージなどの資源を，動的に組合わせて実行環境を構築し，提供する試みが，国内外で複数行われている[1)–4)]．ここで，資源全体から多様なユーザの要求を満たす資源群を確保するには，コアロケーションが重要な課題となる．

我々は，資源が複数の組織から提供されることを前提とし，計算・ネットワーク資源を同時に確保するための，オンラインコアロケーション手法を提案している[5)]．提案手法では，ユーザの計算機及びネットワークに関する要求に対して，各資源を管理する資源マネージャからある時間帯における動的資源情報を入手し，その情報から複数予約プランを作成する．この際のスケジューリングは，最適化問題にモデル化して行う．

また，資源の性能要求に加えて，ユーザの観点では (a) 時刻優先，(b) 価格優先，または (c) 品質優先（可用性など）の方針，資源管理者の観点では，(A) 複数資源マネージャに対して平等に負荷を割り当てる，(B) 省エネや組織間連携のため，特定の資源を優先する，(C) ユーザのサービスレベルを考慮する方針が考えられる．既発表研究の手法では，これらの方針も反映することができる．

このような資源のスケジューリング問題は NP 完全であるため，最適化問題として解くと，実在する資源サイトの数や要求する資源の数など，変数の数の増加によって求解時間が指数的に長くなり，実用化に向けて課題が残る．ILOG 社の CPLEX[6)] は，既発表研究[5)] で用いたフリーの整数計画問題のソルバ GLPK(GNU Linear Programming Kit)[7)] より非常に高速であることが一般に知られている．このような高速な商用ソルバを使うことである程度解消できると考えられる．しかしながら，商用ソルバは非常に高額であり，最適化問題として解くと変数の数に依存して指数的に求解時間が長くなるという問題は依然残る．一方，本研究で対象としているスケジューリング問題では，制約条件を満たすことが重要であり，厳密な最適解を求める必要はない．

また，既発表研究では，予約成功率の評価と，資源管理者の割り当て方針 (C) ユーザのサービスレベルを考慮した場合の評価を行ったが，(A)，(B) に関する評価は行われていない．

本研究では，(1) 制約条件を追加することにより既発表研究の手法を改良し，求解時間の

†1 産業技術総合研究所 National Institute of Advanced Industrial Science and Technology (AIST)

Vol.2009-HPC-121 No.35
2009/8/6

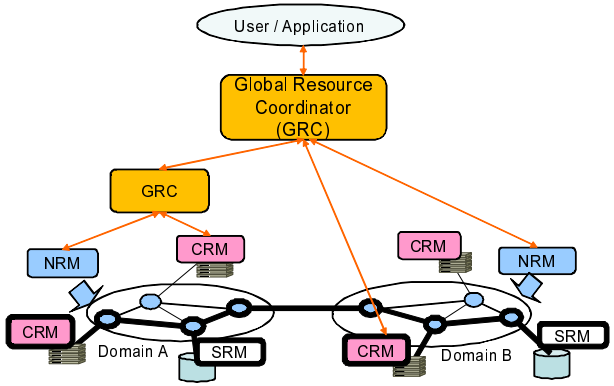

図 1 資源管理フレームワークの概要

短縮を図るとともに，(2) 制約条件とソルバの違いによるに求解時間の比較を行い，実用化に向けて議論する．また，既発表研究の手法に対して追加の実験を行い，(3) 重み付けすることにより管理者の要求を反映した資源の割り当てが可能であり，我々の手法が機能面でも有効であることを示す．

2. コアロケーションモデル

2.1 コアロケーションシステムモデル

我々は，ネットワーク資源を含む多様な資源の管理フレームワーク GridARS の開発している[8]．本フレームワークは資源管理システム (RMS)，資源レジストリ (RR)[9]，資源モニタリングシステム (MS)[10] で構成される．RR は各 RM の所在や各資源の資源マネージャ (RM) の管理する資源の静的な情報を提供し，MS は予約資源の稼働時の状況を提供する．

RMS の概要を**図 1** に示す．RMS は，グローバル資源コーディネータ (GRC) と各資源を管理する資源マネージャ(RM) で構成され，GRC と RM が連携してユーザに資源群を提供する．図 1 の NRM，CRM，SRM は，それぞれネットワーク，計算，ストレージの RM を表す．GRC のうち，スケジューリング機能（プランナ）をもつものが，資源の予約プランを決定する．

2.2 ユーザの資源要求

図 2 にユーザの資源要求（左）と，その要求からプランナが生成する予約プラン（右）の例を示す．計算資源の要求では，計算サイト数，各サイトでの CPU（コア）数，属性情報

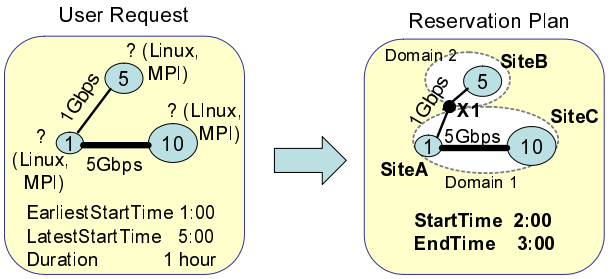

図 2 ユーザの資源要求とプランナによる予約プラン

（OS，実行環境等）を，ネットワーク資源要求では要求する拠点間の帯域，遅延，その他の属性情報 (メディアタイプ，可用性等) を指定する．また，各資源に対して時刻の要求を指定することができる．時刻は，全資源の確保時間帯を直接，または最も早い開始時刻 EST，最も遅い開始時刻 LST，予約時間幅 D により指定する．図 2 右はプランナが選定した予約プラン例で，計算資源 SiteA，SiteB，SiteC とその間の通信経路，資源の確保開始・終了時刻が決定している．通信経路は，抽象化されたトポロジで表わされ，複数ドメインを跨る場合は複数のパスで表される．図 2 では，SiteA-SiteB 間の経路は X1 を経由して Domain1 と Domain2 の 2 つのパスで構成される．

3. コアロケーション手法の改良

3.1 既発表研究におけるコアロケーション手法の概要

GRC における資源確保手順は以下の通りである．

1. 静的資源情報をあらかじめ RR から取得する．
2. ユーザの資源要求を受け取る．
3. GRC のプランナで資源要求に対する予約プランを複数作成する．この際，プランナは利用可能な資源の情報を資源提供候補となる各 RM に問い合わせる．
4. プランナの作成した予約プランから，GRC と複数 RM が連携して資源を確保する．
5. 確保が成功した場合は終了，失敗した場合はその情報をユーザに提供する．

資源を確保する時間帯が直接指定される場合は指定時刻から資源を探索し，時間幅で指定される場合は $[EST, LST + D]$ から資源を探索する．本研究では，商用サービスを考慮して G-lambda プロジェクト[11] で規定している資源予約インタフェース GNS-WSI を用い

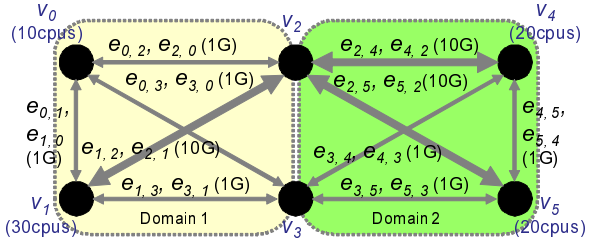

図 3 資源グラフ

る．GNS-WSI では，各 RM の詳細な予約テーブルを公開しない代わりに，指定した時刻における利用可能な資源情報を提供するため，$[EST, LST + D]$ から予約時間帯候補を等間隔に N 個選ぶことにする．選んだ各予約時間帯に対して，次節のコアロケーションアルゴリズムを適用する．

3.2 既発表研究のコアロケーションアルゴリズム

提案手法では，**図 3** に示すような有向グラフ $G = (V, E)$ として資源群を表す．V は G の点の集合，E は G の枝の集合であり，グラフの各点 v_q は計算資源サイトまたはネットワークドメイン間の交換点を，各枝 $e_{+r}(= e_{o,p})$ または $e_{-r}(= e_{p,o})$ は各 NRM が提供する資源間のパスを表す．グラフの点および枝の括弧内は，提供可能な CPU 数および帯域であり，それぞれ $wc_i(i \in V)$，$wb_k(k \in E)$ と表す．v_2，v_3 はドメイン間交換点であり，利用可能な CPU は存在しない．また，提供する CPU および帯域の単位あたりの重み（価格など）を，それぞれ $vc_i(i \in V)$，$vb_k(k \in E)$ と表す．wb_k および vb_k は，e_{+r}, e_{-r} で共有する．

次に，資源要求を完全グラフ $G_r = (V_r, E_r)$ で表す．V_r は必要とする計算資源サイト（点）の集合，E_r は V_r 間を結ぶネットワーク（枝）の集合を示す．また，G_r に必要な CPU 数および帯域をそれぞれ $rc_j(j \in V_r)$，$rb_l(l \in E_r)$ とする．

以上のパラメータから，以下の変数を整数計画法で求めることで予約プランを決定する．

$$x_{i,j} \in \{0,1\} \qquad (i \in V, j \in V_r) \tag{1}$$

$$y_{k,l} \in \{0,1\}\ (k = (m,n) \in E, m,n \in V, l = (o,p) \in E_r, o,p \in V_r) \tag{2}$$

ここで，資源要求されるサイトに対して選択される計算資源サイトの要素 $x_{i,j}$ は 1，その他は 0 となる．また，資源要求されるネットワークに対して選択されるパスの要素 $y_{k,l}$ は 1，その他は 0 となる．次に，目的関数と制約条件は次のように表せる．

$$Minimize$$

$$\sum_{i \in V, j \in V_r} vc_i \cdot rc_j \cdot x_{i,j} + \sum_{k \in E, l \in E_r} ve_k \cdot rb_l \cdot y_{k,l} \tag{3}$$

$$Subject\ to$$

$$\forall j \in V_r,\ \sum_{i \in V} x_{i,j} = 1 \tag{4}$$

$$\forall i \in V,\ \sum_{j \in V_r} x_{i,j} \leq 1 \tag{5}$$

$$\forall i \in V,\ \sum_{j \in V_r} rc_j \cdot x_{i,j} \leq wc_i \tag{6}$$

$$\forall l \in E_r,\ \sum_{k \in E} y_{k,l} \begin{cases} \geq 1 & (rb_l \neq 0) \\ = 0 & (rb_l = 0) \end{cases} \tag{7}$$

$$\forall k \in E,\ \sum_{l \in E_r} rb_l \cdot y_{k,l} \leq wb_k \tag{8}$$

$$\forall l = (o,p) \in E_r, \forall m \in V,$$

$$\sum_{n \in V, m \neq n} y_{(n,m),(o,p)} - \sum_{n \in V, m \neq n} y_{(m,n),(o,p)} = \begin{cases} x_{m,o} - x_{m,p} & (rb_l > 0) \\ 0 & (rb_l = 0) \end{cases} \tag{9}$$

式 (3) の目的関数は，選択する計算・ネットワーク資源の重みの合計を最小化することを表している．式 (4) は，要求サイト j に割り当てられる実際のサイトは 1 つのみであること，式 (5) は，各資源サイト i は資源要求の中で 1 つ以上割り当てられないことを表す．式 (6) は，各資源サイト i が要求される CPU 数以上の CPU を提供可能であることを示している．式 (7) は，要求サイト間のパス l の帯域の確保要求がある場合は，$y_{k,l}$ の総和が 1 以上に，ない場合は 0 になることを表す．式 (8) は，各パス k が要求される帯域以上の帯域を提供可能であることを示す．

式 (9) は，流量保存則[12] を応用して $x_{i,j}$ と $y_{k,l}$ を関連付ける制約を表す．流量保存則では，流量を f とすると，始点の供給量は f，終点の供給量は $-f$，通過点の供給量は 0 となる．よって，流量 $f = 1$ と想定して適用する．すなわち，m が l の始点であれば，式 (9) の値は 1，終点であれば -1，通過点であれば 0 となる．この際，m が始点ならば $x_{m,o} = 1$，

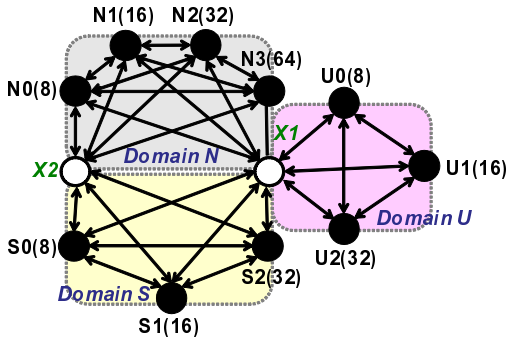

図 4 シミュレーション環境

終点ならば $x_{m,p} = 1$ となるので，式 (9) の値を $x_{m,o} - x_{m,p}$ として表すことができる．

3.3 コアロケーションアルゴリズムの改良点

最適化問題では，求める変数の数が増えると求解時間が指数的に増大する問題がある．よって，3.2 節に以下の 2 つの制約，すなわち枝刈り条件を加えることで，求解時間を短くすることを試みる．

Subject to

$$\forall k \in E, \forall l = (o,p) \in E_r,\ y_{k,(o,p)} + y_{k,(p,o)} \leq 1 \tag{10}$$

$$\forall l \in E_r,\ \sum_{k \in E} y_{k,l} \leq N_{max} \tag{11}$$

式 (10) は，求めるネットワーク（式 (9) におけるフロー）に対して，有向グラフの双方向の枝を同時に選択しないことを表す．これにより，無駄な経路の探索を防ぐことができる．式 (11) は，1 つのネットワークに対して構成する枝の数の最大値 N_{max} を指定するものであり，N_{max} は経験的に与えるパラメータとなる．よって，式 (11) は最適解を求めるわけではないが，探索範囲が大幅に削減できるため，求解時間の短縮には有効であると考えられる．

4. 評　　価

4.1 シミュレーションモデル

評価では，式 (10)，式 (11) の制約条件とソルバの違いによる求解時間の比較と，既発表研究に対して追加の実験を行い，我々の手法によって資源管理者の要求を反映したスケジューリングができることを示す．

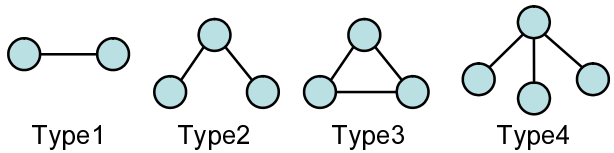

図 5 資源要求の種類

表 1 シミュレーション設定

環境設定	
資源構成	GRC 数=1, NRM 数=3 (N, S, U), CRM 数=10
サイト数/ドメイン名	4/N, 3/S, 3/U
ドメイン交換点	X1 {N, S, U}, X2 {N, S}
CPU 数	N{8, 16, 32, 64}, S{8, 16, 32}, U{8, 16, 32}
CPU 単価	1
帯域 [Gbps]	ドメイン内 = 5, 交換点接続 = 10
帯域単価	ドメイン内 = 5, 交換点接続 = 3
資源要求設定	
ユーザ	UserA, UserB
資源要求の種類	Type1,2,3,4（**図 5**）（一様分布）
要求 CPU 数	各 type の全サイトに対し，1, 2, 4, 8（一様分布）
要求帯域 [Gbps]	各 type の全ネットワークに対し，1（固定）
各ユーザの要求到着間隔	ポアソン到着
予約時間幅 [min]	30, 60, 120（一様分布）
LST - EST(探索時間幅)	平均要求到着間隔 × 3

図 4 に各実験で用いるシミュレーション環境を示す．これは，実際の実証実験で用いた計算資源のサイトとネットワーク[2] を想定し，抽象化したものである．図 4 に示すように，ドメイン内のサイト (黒丸) 間は完全グラフで接続され，ドメイン内の各交換点 (白丸) とドメイン内の全サイトもそれぞれ接続されている．

シミュレーション設定の概要を**表 1** に示す．シミュレーションでは，2 人のユーザが 1 つの GRC に対して資源予約要求をする．資源要求の種類は，図 5 のいずれかを同時に確保するものである．各ユーザにおける資源要求設定は両方とも表 1 に従う．最初の 24 時間までに各ユーザの資源要求が到着し，それぞれ次の 24 時間の時間帯の中から資源の予約を要求をする．2 人のユーザの要求の平均到着間隔は，24 時間の間に計算資源に関して 100[%] の要求率になるように設定した．また，GRC での予約時間帯候補数 N は 10 とし，候補間での優先順位は時間優先とした．

表 2 求解時間の平均値，最大値，最大値／平均値の比較

手法	平均値 [sec]	最大値 [sec]	最大値／平均値
GLPK	0.779	8.492	10.901
GLPK-st	0.333	4.205	12.628
MiniSat-st	12.848	216.434	16.846
MiniSat-st-1	1.918	2.753	1.435

4.2 求解時間の比較

4.2.1 ソルバの比較

求解時間の比較では，既発表研究の手法と改善手法の比較と，求解に用いるソルバの比較を行う．ソルバの比較では，線形計画法（LP）のソルバである GLPK (GNU Linear Programming Kit)[7] と，充足可能性問題（SAT）のソルバ MiniSat[13] と Sugar++[14] を組み合わせたものを用いた．いずれもオープンソースソフトウェアである．

LP は，1 次の不等式または等式のみで目的関数と制約条件が定義された最適化問題のことを呼ぶ．GLPK では，LP を解くアルゴリズムとして単体法（simplex method）と内点法（interior point method）を用いており，デフォルトでは単体法が適用される．本実験においても，単体法を用いている．

SAT のソルバは，乗法標準形（CNF）で与えられた全変数に対して，すべての制約条件が満たされていれば充足可能（SAT），満たされていなければ充足不可能（UNSAT）と判定するものである．我々の手法は最適化問題としてモデル化されているため，SAT のソルバをそのまま利用することはできない．SAT のソルバを用いるため，本研究では Sugar++ を用いた．Sugar++は Sugar[15] を改良したものである．Sugar は，制約充足問題（CSP）を CNP に変換するためのソフトウェアであり[16]，Sugar++は CSP を最適化問題として解くために改良されたものである．Sugar++では，最適解を求めるために最小化または最大化する目的関数の上限，下限を一時的に固定して CSP として解き，複数回 CSP を解くことにより最適解を求める．

4.2.2 求解時間の評価結果

求解時間の評価では，以下の 4 通りの手法を比較した．

GLPK　既発表研究の手法に対して GLPK を用い，最適解を求める．

GLPK-st　改良手法（式 (10)，式 (11) を追加）に対して，GLPK を用い，最適解を求める．

MiniSat-st　改良手法に対して，Sugar++と MiniSat を用い，最適解を求める．

MiniSat-st-1　改良手法に対して，Sugar++と MiniSat を用い，制約条件を満たす解の 1 つを求める (SAT の計算回数は 1 回)．

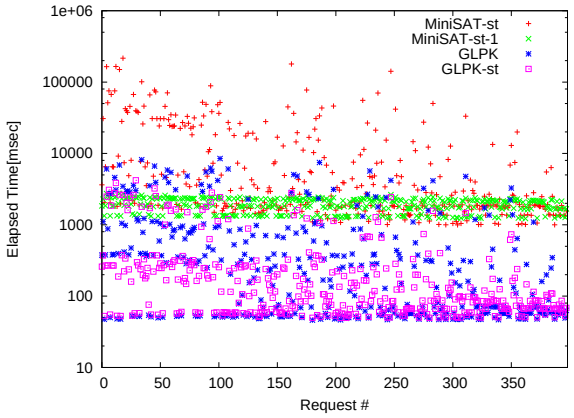

図 6　各資源要求における求解時間の比較（ログスケール）．

求められる解の質は，GLPK ≥ GLPK-st = MiniSat-st > MiniSat-st-1 となる．シミュレーションは，いずれも CPU Intel Core2 Quad Q9550(2.83GHz)，OS CentOS 5.0, kernel 2.6.18 x86_64, メモリ 4GB の計算機上で実行した．

表 2 に，各手法における求解時間の平均値，最大値と最大値／最小値の割合を示す．GLPK と GLPK-st で比較すると，平均値，最大値のいずれも GLPK-st の方が 2 倍以上短くなっていることが分かる．このことから，最適化問題に制約条件を加えることで求解時間を短縮する手法は，非常に有効であることが分かる．

次に，同じ制約条件を使った GLPK-st と MiniSat-st を比較すると，GLPK-st の方が圧倒的に速い結果となった．このことから，本研究のコアロケーションの問題の最適化問題のソルバとしては，GLPK の方が適していることが分かる．一方，求解時間の最大値および最大値／平均値の値ですべての手法を比較すると，MiniSat-st-1 が一番小さいという結果が得られた．

図 6，**図 7** に各資源要求における求解時間の結果を示す．横軸は要求された資源要求の順序を表し，各プロットは各要求に対して予約候補数 N =10 回の計算時間の平均値をプロットしたものである．この N 個の計算は独立して計算できるため，実環境で利用する場合において実際にかかる時間を想定して比較した．

図 6 は y 軸をログスケールで表したものであり，図 7 はそのうちの 0〜10[sec] までの結果を出力したものである．グラフから，最適解を求める手法では，いずれも問題によって求

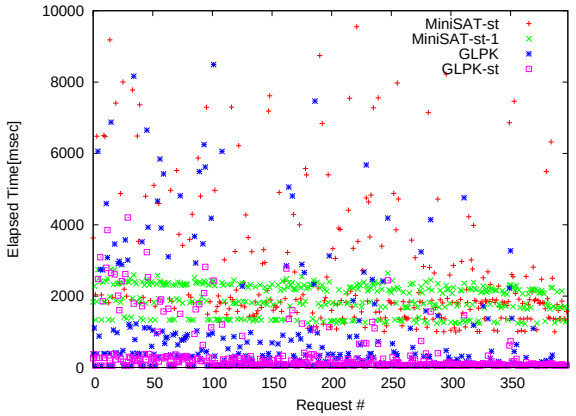

図 7 各資源要求における求解時間の比較（0〜10[sec] まで結果）.

解時間に大きなばらつきがあるのに対し，MiniSat-st-1 の結果ではばらつきはみられない．また，図 7 では，MiniSat-st-1 の結果を見ると 3 本の筋が確認できる．これは図 5 の点の数の違いによるものである．すなわち，最適解を求めない手法においては，多項式時間で求解できることが分かる．

また，最適解を求める手法の結果を見ると，いずれも要求数が多くなるほどばらつきが減っていくことが分かる．これは利用可能な資源の数が少なくなるにつれて，最適解の探索範囲が狭まるためである．

4.2.3 議　　論

従来のスケジューリングに関する研究では，NP 完全であるために様々な複雑かつヒューリスティックな手法が提案されてきた．一方で，近年の計算機性能の向上と LP ソルバの研究成果により，LP の実求解時間は短縮されている．また，LP は制約条件を加えていくほど求解時間が短くなるという性質を持つ．CPLEX などの商用ソルバでは，前処理で制約条件を加えることにより，求解時間の短縮を図っているといわれている．また，最適解を求めなければ多項式時間で求解できるため，その点でも求解時間の性能改善の余地がある．

提案手法では，LP で解くとネットワークを完全グラフで表現しなければならないため，サイト数の 2 乗のオーダーで求解時間が長くなる可能性がある．しかしながら，以下のような性質もある．

- 図 1 で示したように，プランニングするスケジューラ（GRC）が階層的に構成されており，探索範囲の局所化が可能である．
- モデル化した際の点の数が，計算機の台数ではなくサイトの数でスケールする．
- ネットワークのレイテンシや計算の実行環境など，今後様々な要求を受け入れることを前提としており，さらに制約条件を加えることが可能である．

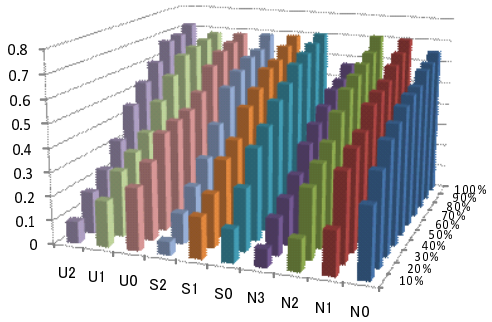

図 8 サイトの負荷の比較（(A)．重み付けなし）.

よって，本研究のスケジューリング問題に対して，LP を適用することは有効なアプローチであると考えられる．

4.3 資源管理者の観点に対する提案手法の有効性の評価

次に，既発表研究に対して追加の実験を行った．本実験では，我々の手法により資源管理者の要求が反映できることを示す．

資源管理者の観点では，(A) 複数 RM に対して平等に負荷を割り当てる，(B) 省エネや組織間連携のため，特定の資源を優先する，(C) ユーザのサービスレベルを考慮する方針が考えられる．(C) は既発表研究において示したため，本研究では，(A)，(B) に関してシミュレーションした結果を示す．ここで、(B) では CPU の単価に対して重み付けすることにより、特定の資源を優先するようにした．

図 8 に (A) の方針，**図 9**，**図 10** に (B) の方針でシミュレートした際の各サイトの平均負荷を示す．図 9 は計算サイトの所有する CPU 数ごとに 64：32：16：8 = 1：10：100：1000 のように重み付けし，大きいサイトに優先的に割り当てられるようにした．図 10 は，ネットワークドメインごとに N：S：U = 1：10：100 のように重み付けし，N，S，U の順に．優先的に割り当てられるようにした．

Vol.2009-HPC-121 No.35
2009/8/6

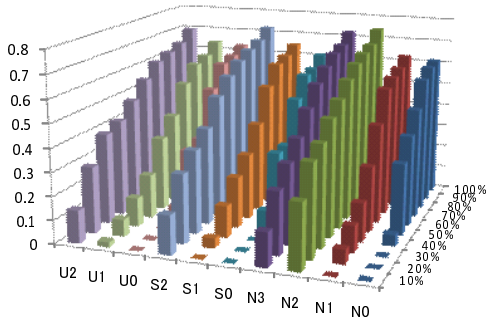

図 9 サイトの負荷の比較（(B)．サイト CPU 数ごとに重み付け）.

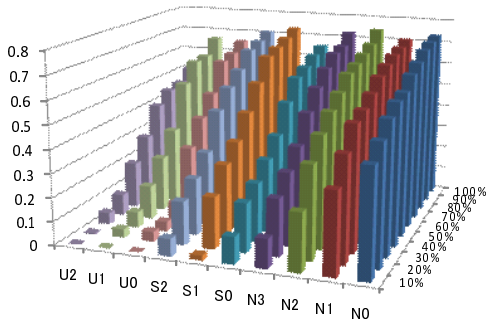

図 10 サイトの負荷の比較（(B)．ネットワークドメインごとに重み付け）.

シミュレーション結果から，図 8 はすべてのサイトにおいてほぼ均一に負荷が上がっているのに対し，図 9 では CPU 数の多いサイトに，図 10 では，N, S, U の順に資源が優先的に割り当てられていることが分かる．一方，シミュレーション時間が長くなると資源が少なくなるためにいずれの結果も負荷が均一になっていく．よって，我々の手法に重み付けをすることにより，資源が豊富にある状況では資源管理者の要求を反映することができることが分かった．

5. 関 連 研 究

安藤ら[17] は，本研究同様，複数計算・ネットワーク資源を確保するための予約プランを生成する予約アルゴリズムを提案している．提案するアルゴリズムは資源群を無向グラフで表し，バックトラック法により資源群を探索する．本研究ではコアロケーションのための手法を提案しているが，安藤らのアルゴリズムではワークフローにも対応する．また，バックトラック法でインクリメンタルに予約手続きを行う手法でよい性能を示しているが，手続きが複雑になり，多くの資源をブロックしてしまうという問題もある．

Roblitz も最適化問題に基づいて複数資源のコアロケーションするためのスケジューリングアルゴリズムを提案している[18]．この研究も，コアロケーションとワークフローの要求に対するスケジューリングを行う．この研究では，ネットワークのトポロジは想定しない，ソルバに CPLEX を用いているという点で異なる．

6. ま と め

本研究では，既発表研究で提案した性能を保証する計算・ネットワーク資源のオンラインコアロケーション手法に対して，制約条件を追加することにより LP の求解時間の短縮するとともに，制約条件とソルバの違いによるに求解時間を比較した．現在の想定環境でも実用レベルであり，また将来的にサイト数がスケールする状況においても，求解時間の短縮の余地があるため，我々の手法が実用的であることが示された．また，我々の手法に対して重み付けすることにより，管理者の要求を反映した資源の割り当てが可能であり，機能面での有効性も示した．

今後は，LP の求解時間の短縮に関してさらに検討していくとともに，より様々な条件を反映できるように改良していく．また，我々の開発する資源管理システム GridARS への組み込みを行い，提案手法を実用化する．

謝辞 Sugar を利用するにあたり，貴重なご意見を下さった神戸大学の田村直之先生，Sugar++で MiniSat を用いて最適化問題を解くために必要な未公開パッチを提供してくださった丹生智也様に感謝いたします．また，最適化問題に関してご助言いただいた中央大学の藤澤克樹先生，安井雄一郎様に感謝いたします．

本研究の一部は，科研費（21700047）および情報通信研究機構（NICT）の委託研究「ダイナミックネットワーク技術の研究開発」の助成を受けたものである．

参 考 文 献

1) Takefusa, A., Hayashi, M., Nagatsu, N., Nakada, H., Kudoh, T., Miyamoto, T., Otani, T., Tanaka, H., Suzuki, M., Sameshima, Y., Imajuku, W., Jinno, M., Takigawa, Y., Okamoto, S., Tanaka, Y. and Sekiguchi, S.: G-lambda: Coordination of a Grid Scheduler and Lambda Path Service over GMPLS, *Future Generation Computing Systems*, Vol.22(2006), pp.868–875 (2006).
2) Thorpe, S.R., Battestilli, L., Karmous-Edwards, G., Hutanu, A., MacLaren, J., Mambretti, J., Moore, J.H., Sundar, K.S., Xin, Y., Takefusa, A., Hayashi, M., Hirano, A., Okamoto, S., Kudoh, T., Miyamoto, T., Tsukishima, Y., Otani, T., Nakada, H., Tanaka, H., Taniguchi, A., Sameshima, Y. and Jinno, M.: G-lambda and EnLIGHTened: Wrapped In Middleware Co-allocating Compute and Network Resources Across Japan and the US, *Proc. GridNets2007* (2007).
3) Barz, C., Pilz, M., Eickermann, T., Kirtchakova, L., Waldrich, O. and Ziegler, W.: Co-Allocation of Compute and Network Resources in the VIOLA Testbed, TR 0051, CoreGrid (2006).
4) : AAA scenarios and test-bed experiences, Deliverable reference number:d4.2, The PHOSPHORUS project (2008). http://www.ist-phosphorus.eu/files/deliverables/Phosphorus-deliverable-D4.2.pdf.
5) 竹房あつ子，中田秀基，工藤知宏，田中良夫：高品質分散実行環境のための計算・ネットワーク資源のグローバルスケジューリング手法，情報処理学会研究報告 2008-HPC-119，pp.55–60 (2009).
6) ILOG CPLEX: http://www.ilog.co.jp/product/opti/cplex/cplex.html.
7) GLPK (GNU Linear Programming Kit): http://www.gnu.org/software/glpk/glpk.html.
8) 竹房あつ子，中田秀基，工藤知宏，田中良夫，関口智嗣：多様な資源を事前予約で同時確保するためのグリッドコアロケーションシステムフレームワーク GridARS，情報処理学会論文誌コンピューティングシステム (ACS20)， Vol.48, No.SIG18, pp.32–42 (2007).
9) 小島功，木本正裕，的野晃整：データ型別索引に基づく異種情報の検索システムとそれを用いた地球観測情報レジストリの実現，情報処理学会等「データ工学と情報マネジメントに関するフォーラム 2009」 (2009). http://db-event.jpn.org/deim2009/proceedings/files/B2-4.pdf.
10) Takefusa, A., Nakada, H., Yanagita, S., Okazaki, F., Kudoh, T. and Tanaka, Y.: Design of a Domain Authorization-based Hierarchical Distributed Resource Monitoring System in cooperation with Resource Reservation, *Proc. HPC Asia 2009 (in print)* (2009).
11) G-lambda プロジェクト：http://www.g-lambda.net/.
12) Ahuja, R.K., Magnanti, T.L. and Orlin, J.B.: *Network Flows: Theory, Algorithms, and Applications*, Prentice Hall (1993).
13) MiniSat: http://minisat.se/.
14) Tanjo, T., Tamura, N. and Banbara, M.: Sugar++: A SAT-based Max-CSP/COP Solver, *Proc. the Third International CSP Solver Competition*, pp.144–151 (2008).
15) NaoyukiTamura, T.T. and Banbara, M.: System Description of a SAT-based CSP Solver Sugar, *Proc. the Third International CSP Solver Competition*, pp. 71–75 (2008).
16) Tamura, N., Taga, A., Kitagawa, S. and Banbara, M.: Compiling Finite Linear CSP into SAT, pp.254–272 (2009).
17) 安藤誠士郎，合田憲人：グリッド上の事前予約スケジューリング手法の性能評価，情報処理学会研究報告 2007-HPC-113，pp.37–42 (2007).
18) Roblitz, T.: Global Optimization For Scheduling Multiple Co-Reservations In The Grid, *Proc. CoreGRID Symposium*, pp.93–109 (2008).