数理解析研究所講究録
第1676巻 2010年 16-27

半正定値計画問題に対するソフトウェア開発で用いられる新技術について

New Technologies in the SDPA Project

藤澤 克樹 (Fujisawa Katsuki)
中央大学理工学部経営システム工学科
Department of Industrial and Systems Engineering, Chuo University

Abstract:

The SDPA Project started in 1995 have provided several software packages for solving large-scale Semidefinite Programs(SDPs). Further improvements are necessary for the software packages since optimization problems become larger and more complicated. We show some current works and new technologies in the SDPA project as follows; (I) The memory hierarchy is carefully considered to specify the bottleneck of the algorithm and improve the performance. The latest version of the SDPA supports the multi-thread computing on multi-core processor, and solves large-scale SDPs quickly and efficiently. (II) We have developed a web portal system utilizing the cloud computing technology for some software packages in the SDPA Project.

Key words: high-performance computing, parallel computing, optimization software, semidefinite programming

1 情報技術の発展と最適化

計算機や通信技術はその登場以来、常に進化を遂げていることは良く知られているが、最適化などの様々の分野において、想定する問題の規模、適用する手法、実問題への適用可能性などの最適化問題に対するパラダイムに大きな変化が生じるようになった [8]. ただし計算量の理論から考察すると $\mathcal{NP}$-困難な問題は問題の大きさが増加するにつれて、必要な計算量が指数関数的に増大するので、ある程度大きな規模では最新の情報技術を用いても最適解を求めることは依然として極めて難しい. しかし近年では主に以下の二つの方法やそれらの方法の組合せによって最適化問題を解く研究がさかんに行われており、アルゴリズムの発展や並列計算の適用などによって目覚しい成果が達成されている.

1. 超高速、大容量の計算機を集めて従来より提案されている手法に並列計算の技術を適用して問題を解く. ただし様々な理論的成果を採り入れて、計算時間を減らす工夫を採用している.
2. 近似解法を適用して実用的な時間で優れた近似解を得ることを目指す.

まず1は巡回セールスマン問題 (TSP) に対する分枝カット法や二次割当問題に対する分枝限定法などが有名である. 2009年現在では 85,900 点を持つ大きな対称 TSP の最適解が求められたことが報告されている[1]. 後者は平均で 653 個、最大で 1,007 個の CPU

[1] http://www.tsp.gatech.edu/

を用いて約1週間で30次元のNUG30という問題の最適解を求めた．また著者らのグループが数万制約，数百万非零要素を持つ 半正定値計画問題 (以下 SDP) を解くことに成功した [7].

また2は配送計画問題などの組合せ最適化問題に対して近傍探索などを並列化したメタ解法を適用して問題を解く例などが最近では多く見られる．

さらに数理計画問題を解くためのソフトウェアもアルゴリズムの改善や計算機能力の向上によって大幅に性能を向上させている．数理計画問題は問題の大きさ n の多項式時間で解けるものも多いが，想定される数理計画問題が巨大であるときにはCPU単体のみの性能向上だけでは短時間に問題を解くことは困難である．それらの困難を克服するための並列計算の手法として**クラスタ計算 (Cluster Computing)** と**クラウド計算 (Cloud Computing)** 及び**グリッド計算 (Grid Computing)** などが最近注目を集めている．また最近ではマルチコア・プロセッサが主流になりつつあり Intel Core 2, Core i7 系や PlayStation3 などに搭載されている Cell (IBM/TOSHIBA/SCEI) などがある．

表 1: SDPA 7.3.1 と SDPA 2.0.1 の比較実験：mcp500-1.dat-s

SDPA 7.3.1(2009年)	SDPA 2.0.1(2009年)	SDPA 2.0.1(1996年)
1.3 秒 (8 コア), 4.1 秒 (1 コア)	504.7 秒	133,892.5 秒

1996年：SONY NEWS-5000WI, CPU MIPS R4400 133MHz, memory 128MB
2009年：Dell PowerEdge R710, CPU Intel Xeon X5550 2.66GHz, memory 72GB

表1はSDPを解くためのソフトウェアSDPA [2]の比較実験の結果を示したものである．SDPAは著者らのグループが開発したソフトウェアで1995年からインターネットを通じて公開を行っている．表1には1996年にリリースされたSDPA 2.0.1用いてmcp500-1.dat-s(SDPのベンチマーク問題集SDPLIB [3] に収録されている) SDPを解いた場合の結果が含まれているが，当時のワークステーション (SONY NEWS) で133,892.5秒 (およそ37.2時間) の時間を要した．またSDPA 2.0.1を現在のワークステーション (Dell PowerEdge R710) で実行してmcp500-1.dat-sを解いた場合には504.7秒で解いている．よって粗く計算すれば1996年から2009年にかけて計算機の進歩によりソフトウェアが $133{,}892.5 / 504.7 \approx 265.3$ 倍高速化されていることがわかる．また最新のSDPA 7.3.1でmcp500-1.dat-sを解いた場合で1.3秒で解くことができるので，アルゴリズムによる高速化も同様に計算すると $504.7 / 1.3 \approx 388.2$ 倍高速化されていることになるので計算機の高速化だけでなくアルゴリズム等の高速化も主因の一つであると考えることができる．このようにソフトウェアの高速化には計算機やアルゴリズムの高速化など様々な要因が関係していることがわかる．

2 最適化ソフトウェアの実装方式

近年，アルゴリズムサイエンス分野における基礎理論の探求は飛躍的に進んでおり，以前のような理論的計算量を重視する立場から，ソフトウェア実装面を意識して，実際にどのような構造，特性を持つ場合において，高速かつ安定に解けるかといった研究に移行し

[2] http://sdpa.indsys.chuo-u.ac.jp/sdpa/
[3] http://www.nmt.edu/~borchers/sdplib.html

てきている．一方，コンピュータのハードウェア＆ソフトウェア面での研究の進展も著しく，近年ではスーパーコンピュータ等を中心とした並列計算技術だけでなく，汎用的なマルチコア上の並列計算から大規模環境下でのクラスタやクラウド計算などの新世代の実装方式が開発されている．多くの最適化問題において高速＆高性能化が達成されていることに疑いはないのだが，例えば

1. 具体的にコンピュータ内部（ハード＆ソフト）でどのような現象が起きているのか？
2. どこまで性能を上げることが出来るのか？（理論的限界等）
3. 実用的な観点からハードの仕様や実行時間に制限が付いたら対応は可能か？

といった質問に対して，現在の複雑かつ高度な数理科学と情報科学の発展の中で，説得力を持った答えを見付けることは困難である．そこで，数理科学分野と情報科学分野の最新の成果を持ち寄り，アルゴリズムサイエンスの立場における最適化ソフトウェアの実装方式について解明を行っていくことが必要とされている．具体的には以下の方針や技法，結果などを重視する方針を取っている．

1. 理論的限界等から性能目標を定める
2. トレードオフ関係の把握
3. 計算量とデータ移動量の正確な推定を行う
4. データの特性 (疎性，サイズ) と性能値の関係を見極める

著者らのグループではすでに述べたように代表的な数理計画問題である SDP に対する内点法を記述したソフトウェア SDPA の開発・評価・公開を１０年以上行っている [4]．最近では以下のような開発方針を採用している．

1. 近年のソフトウェアにおいては疎データの扱い，疎構造の記述方法が重要になっているが，入力問題に対して自動的に適切な前処理を行い，密データ構造と疎データ構造のどちらで処理するかについて判断を行う．
2. 通常，ソフトウェアのボトルネックは CPU コア内部 (浮動小数点演算能力や L2 キャッシュの帯域幅など) に存在するか，CPU とメモリ間の帯域幅に因るものかのどちらかである場合が多い．この特性を用いて，アルゴリズムの各部分のボトルネック構造を解明することによって適切なアルゴリズムを選択することができる．
3. 数値精度が問題になる場合には，任意精度演算などを用いて得られる精度と計算時間とのトレードオフなどに注目しながら，適切な方法を選択していく．

内点法アルゴリズムにおいては，探索方向を求めるために Schur complement と呼ばれる行列の生成とこの行列の Cholesky 分解を行う必要があり，大きなアルゴリズム的ボトルネックになっている．この部分の並列化と大規模な SDP に対する挑戦については次節にて説明を行う．

[4] http://sdpa.indsys.chuo-u.ac.jp/sdpa/

3 超大規模 SDP に対する数値実験

SDPは幅広い分野に応用を持ち,線形計画問題を対称行列の空間へ拡張した構造を持っていることもあって 21 世紀の線形計画問題としての期待も大きい. 最近, SDPの新たな応用として,物理化学・化学物理において分子の電子構造を求める研究がされている.分子の電子状態は固有値問題であるSchrödinger方程式を解くことによって求めることができる.特に,分子の最も安定した基底状態に相当するのがSchrödinger方程式の最小固有値解であり,その最小固有値は**基底状態エネルギー**と呼ばれる. 分子の基底状態エネルギーは化学反応モデル等の理論的なエネルギー予測に役立つ基本的な値である. 従来のアプローチではSchrödinger方程式の最小固有値解を近似した波動関数から基底状態エネルギーを算出するHartree-Fock (HF) 法, SDCI法, CCSD(T)法等などが広く用いられている. 波動関数を用いるアプローチの代用として1960年代にColeman[1],Garrod-Percus[4]に提案されたのが**縮約密度行列法**と呼ばれる手法である. 1970年代には最も小さな原子等に対して数値実験が行われてきたが,その複雑さや計算の大変さからこの方法自体が忘れられた一面もある. 縮約密度行列法では分子の基底状態が2次の縮約密度行列で表され,その行列の線形制約式の元で最適化問題を解くことによって基底状態エネルギーの下界値が求まる仕組になっており,この最適化問題がSDPとなることが知られている.このSDPの最適変数に相当するのが2次の縮約密度行列であり,目的関数では分子を構成する原子の種類や幾何構造等の情報を持った線形関数を最小化する. 制約式は対象とする特定の分子には依存しない一般の分子の状態を表す N-representability 条件の一部から構成されている. これらは前述のように2次の縮約密度行列の線形結合行列が半正定値行列になるような制約であり,代表的なものに P,Q,G 条件 [1, 4] があり,その他にも $T1$,$T2$ 条件等 [2, 12] が知られている. 縮約密度行列法に登場する問題は2001年に初めて中田真秀ら [9] によってSDPになることが確認され,非常に小さな原子・分子に対してHF法に劣らぬ良い基底状態エネルギーを算出することが分っている. この論文では基本的な P,Q,G 条件を使っているが,[12] ではさらに $T1$,$T2$ 条件をSDPの制約式に追加して,Gaussian 98 等の商用ソフトで提供されている最も優れたCCSD(T)法よりも殆どの場合良い基底状態エネルギーの値を出している. また,縮約密度行列法では従来の方法よりも基底状態エネルギーの他に双極モーメントがより正しく求まることも確認されている [3, 9, 12].

また縮約密度行列法からSDPとして定式化される問題は超大規模になりスーパーコンピュータまたは大型クラスタ計算機級のメモリ量が必要とされるので,SDPに対するアルゴリズムも並列も処理されることが望まれる. 今後SDPの効率的な解法の進歩もしくは高精度の基底状態エネルギーを要求する応用例の出現により,注目されるべき分野であろう.詳しくは [7] を参照のこと.

次にSDPに関する諸定義を行なう. $\Re^{n\times n}$ を $n\times n$ の実行列の集合, $\mathcal{S}^n$ を $n\times n$ の実対称行列の集合とする. 任意の $\boldsymbol{X}$, $\boldsymbol{Z}\in\Re^{n\times n}$ に対して $\boldsymbol{X}\bullet\boldsymbol{Z}$ は $\boldsymbol{X}$ と $\boldsymbol{Z}$ の内積,すなわち Tr $\boldsymbol{X}^T\boldsymbol{Z}$ ($\boldsymbol{X}^T\boldsymbol{Z}$ の trace : 固有和) を表す. $\boldsymbol{X}\succ\boldsymbol{O}$ は $\boldsymbol{X}\in\mathcal{S}^n$ が正定値,つまり任意の $\boldsymbol{u}(\neq 0)\in\Re^n$ に対し $\boldsymbol{u}^T\boldsymbol{X}\boldsymbol{u}>0$ であることを示している. また $\boldsymbol{X}\succeq\boldsymbol{O}$ は $\boldsymbol{X}\in\mathcal{S}^n$ が半正定値,つまり任意の $\boldsymbol{u}\in\Re^n$ に対して $\boldsymbol{u}^T\boldsymbol{X}\boldsymbol{u}\geq 0$ であることを示している. このとき (1) を SDP の等式標準形と呼ぶことにする.

$$
\begin{array}{ll}
\text{主問題：} & \\
\text{最小化} & \boldsymbol{A}_0 \bullet \boldsymbol{X} \\
\text{制約条件} & \boldsymbol{A}_i \bullet \boldsymbol{X} = b_i \quad (i = 1, 2, \ldots, m) \\
& \boldsymbol{X} \succeq \boldsymbol{O} \\
\text{双対問題：} & \\
\text{最大化} & \displaystyle\sum_{i=1}^{m} b_i y_i \\
\text{制約条件} & \displaystyle\sum_{i=1}^{m} \boldsymbol{A}_i y_i + \boldsymbol{Z} = \boldsymbol{C} \\
& \boldsymbol{Z} \succeq \boldsymbol{O}
\end{array}
\tag{1}
$$

表 2: 用語等の定義

記号	定義
n	SDP の行列 $\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{A}_i (i = 0, \ldots, m) \in \mathcal{S}^n$ などの大きさ
m	SDP の主問題における制約式の数
#nonzero	定数行列 $\boldsymbol{A}_i (i = 0, \ldots, m)$ における非零要素の合計数
ELEMENTS	Schur complement 行列 $\boldsymbol{B}$ の全要素の計算に要する計算時間：$\mathcal{O}(mn^3 + m^2n^2)$
CHOLESKY	Schur complement 行列 $\boldsymbol{B}$ の Cholesky 分解：$\mathcal{O}(m^3)$

この節では SDP に対する様々な数値実験結果を紹介していくが、最初にこの節で使用する用語等について定義を行う (表 2). SDP の問題の大きさは n, m と #nonzero で表現することが可能であるが、SDPA [7] ではブロック対角な行列のデータ構造およびそれらの内部演算を組み入れているので、行列の大きさ n については**ブロック対角行列**で考える必要がある. 次にブロック対角行列の一般形を示す. ブロック数とブロック構造ベクトルを用いて，定数行列 $\boldsymbol{A}_i$ や 変数行列 $\boldsymbol{X}$、$\boldsymbol{Z}$ に共通なブロックデータ構造を表現する.

$$
\left.
\begin{array}{ll}
\boldsymbol{F} = \begin{pmatrix} \boldsymbol{G}_1 & \boldsymbol{O} & \boldsymbol{O} & \cdots & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{G}_2 & \boldsymbol{O} & \cdots & \boldsymbol{O} \\ \cdot & \cdot & \cdot & \cdots & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{O} & \boldsymbol{O} & \cdots & \boldsymbol{G}_\ell \end{pmatrix}, & \\
\boldsymbol{G}_i : \ p_i \times p_i \text{ 対称行列} & \\
\quad (i = 1, 2, \ldots, \ell). &
\end{array}
\right\}
$$

SDPA では、定数行列 $\boldsymbol{A}_i$ や変数行列 $\boldsymbol{X}$、$\boldsymbol{Z}$ などを上記にようなデータ構造で保持している. このデータ構造の採用によって入力行列が疎行列や部分的に対角行列を含む場合にも効率の良いデータの保持と計算が可能になった. ある行列を入力したときにその行列のブロック対角構造を自動的に判別するのは時間がかかるので SDPA では以下のように nBLOCK や bLOCKsTRUCT などのパラメータを用いてユーザーがブロック対角構造を指定するようになっている.

$$
\begin{aligned}
\text{nBLOCK} &= \ell, \\
\text{bLOCKsTRUCT} &= (\beta_1, \beta_2, \ldots, \beta_\ell),
\end{aligned}
$$

$$\beta_i = \begin{cases} p_i : \boldsymbol{G}_i \text{ が通常の対称行列のとき} \\ -p_i : \boldsymbol{G}_i \text{ が対角行列のとき} \end{cases}$$

次に下の行列 $\boldsymbol{F}$ を例に用いる.

$$\boldsymbol{F} = \left(\begin{array}{cccccc|ccc} 11.0 & 0 & -13.1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 25.9 & 0 & 0 & 0 \\ -13.1 & 0 & 33.2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 25.9 & 0 & 0 & 6.0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 3.0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1.0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -2.5 \end{array}\right)$$

この場合では $n = 8$ であるが,ブロック対角行列で考えると nBLOCK = 2, bLOCKsTRUCT = $(5, -3)$ である.

今回,数値実験として表 3 のような系に対して SDP 緩和を生成し, SDPARA 7.0.1[10] で解いた. SDPARA とは SDP に対する主双対内点法を実装した SDPA 6.2.1[11] のボトルネックとなっている探索方向の計算のための線形方程式系の計算を MPI や ScaLAPACK などを用いて並列化したソフトウェアである. ただし, $2K$ は基底関数の数, N は電子数, m は SDP の等式制約の本数を示している. ただし表記スペースの問題から bLOCKsTRUCT は大きい順に並べて一部省略してある. 例えば 4,032×2 とは 4,032×4,032 のブロックが二つあるという意味である. -430 とは対角ブロックの大きさが 430 という意味である. 表 3 で最も巨大な SDP は H_2O に関するもので, 図 1 のような巨大ではあるが疎なブロック対角構造を有している. 15,914 × 15,914 の大きさを持つ行列が問題内に 27,888 個存在していて, これらの行列の非零要素の合計数は 4,766,902 個に達する. これまでに解くことができた一般の SDP では世界最大規模であると思われる.

表 3: 各問題の大きさに関するデータ

化学式	$2K$	N	m	n	#nonzero	nBLOCK	bLOCKsTRUCT
LiOH(PQG)	22	12	10,593	1,264	75,690	14	(242,121×4,...,-274)
HF(PQG)	24	10	15,018	1,498	105,694	14	(288,144×4,...,-322)
NH(T1T2)	24	8	15,018	10,170	2,205,558	22	(2,532×2,792×4,...,-322)
BH_3O(T1T2)	26	16	20,709	12,828	3,290,622	22	(3,224×21,014×4,...,-374)
H_2O(T1T2)	28	10	27,888	15,914	4,766,902	22	(4,032×21,274×4,...,-430)

表 3 の中の LiOH, HF, NH, BH_3O に関しては産業技術総合研究所の AIST Super Cluster(ASC) の P32 クラスタで, H_2O に関しては M64 クラスタを用いた. ASC は全体で 14.60TFlops の総演算性能と 9.78TFlops の実効性能を持っており, 全部で三つのクラスタ (P32, M64, F32) と 3,208 個のプロセッサより構成されている. このクラスタの主目的の一つはグリッドに計算能力を提供する基盤システムとして利用し, 高性能なグリッド計算環境を構築することにある. P32 クラスタはメモリ量はそれほど多くないが同時に多くの CPU を必要とする場合に用いられ M64 クラスタは 1CPU あたりのメモリ使用量が極めて大きいときに用いる. また P32 で同時に使用する CPU 数を変えながら問題を解くのに要した計算時間をまとめたのが表 4 である. ただし LiOH, HF は PQG, NH,

表 4: 大規模SDPに対するSDPARAの実験結果1 (単位は秒)

問題名		1台	4台	8台	16台	64台	128台	256台
LiOH (PQG)	総計算時間	14,250	3,514		969	414		
	ELEMENTS	6,150	1,654		308	84		
	CHOLESKY	7,744	1,186		357	141		
HF(PQG)	総計算時間	47,483	8,939		2,549	1,120		
	ELEMENTS	16,719	4,390		706	237		
	CHOLESKY	20,995	3,395		983	331		
NH(T1T2)	総計算時間					66,015	37,028	24,499
	ELEMENTS					47,416	19,958	9,875
	CHOLESKY					820	362	285
BH_3O(T1T2)	総計算時間					148,387		
	ELEMENTS					104,745		
	CHOLESKY					1,989		
H_2O(T1T2)	総計算時間			2,060,237				
	ELEMENTS			1,985,337				
	CHOLESKY			22,137				

表 5: 大規模SDPに対するSDPARAの実験結果2(単位は秒)

問題名		1台	2台	4台	8台	16台	32台	64台	128台
LiOH (PQG)	総計算時間	4,464	1,827	952	521	266	200	189	241
	ELEMENTS	2,211	1,000	506	247	120	68	54	57
	CHOLESKY	2,130	618	328	188	97	72	93	66
HF(PQG)	総計算時間	11,659	4,768	2,297	1,230	640	447	402	435
	ELEMENTS	5,906	2,730	1,249	601	303	179	132	143
	CHOLESKY	5,819	1,622	819	454	238	175	183	133
NH(T1T2)	総計算時間						20,356	12,790	19,680
	ELEMENTS						16,266	9,408	14,890
	CHOLESKY						243	252	217
BH_3O(T1T2)	総計算時間							43,067	
	ELEMENTS							34,496	
	CHOLESKY							603	
H_2O(T1T2)	総計算時間							94,479	
	ELEMENTS							81,296	
	CHOLESKY							787	

BH_3O, H_2O は T1T2 条件である. このため, HF と NH は $2K$ が同じだが, この表の SDP では変数行列のサイズが異なっている. P32 クラスタ の各 PC は CPU として Opteron 246 (2.0GHz) を 2 個搭載し, メモリは 6GB である. また M64 クラスタ の各 PC は CPU として Itanium 2 (1.3GHz) を 4 個搭載し, メモリは 16GB である. なお LiOH, HF は 64 台で十分高速に計算できたため 128 台以上で計算しておらず, 逆に NH では計算時間がかかり過ぎるために 32 台以下で計算していない. また BH_3O では 1CPU あたりのメモリ使用量が 6.4GB なので合計メモリ使用量は 6.4 × 64 台 = 409.6GB にも達する. また H_2O は 1CPU あたりのメモリ使用量が 11.2GB なので合計メモリ使用量は 11.2 × 8 台 = 89.6GB にも達し, 実行時間は約 24 日である. 全ての数値実験において得られた解の精度も良く, この規模で精度の良い解が求まるのは前例の無いことである. SDPARA はこのような超大規模な SDP の計算において他のソフトウェアの追随を許していない.

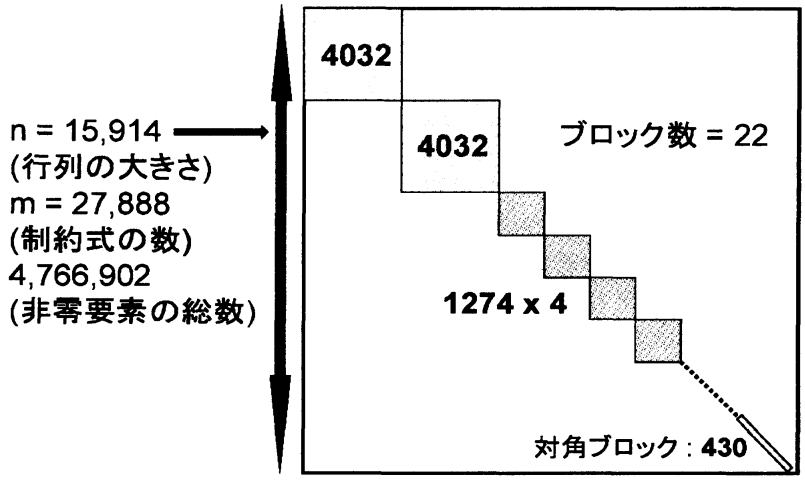

図 1: 超巨大 SDP のブロック対角構造

表 6: 各クラスタ計算機の構成や性能

	SDPA クラスタ	M64 クラスタ	P32 クラスタ
CPU	Intel Xeon 5460 (3.16GHz)	Intel Itanium 2 Xeon (1.3GHz)	AMD Opteron 246 (2.0GHz)
メモリ	48 GByte	16 Gbyte	6 Gbyte
NIC	Myrinet-10G × 1	Myrinet-2000 × 1	Myrinet-2000 × 1
合計数	16 ノード 32CPU(128 コア)	132 ノード 528CPU(528 コア)	1072 ノード 2,144CPU(2,144 コア)
LINPACK	1.43 TFlops	1.62 TFlops	6.16 TFlops

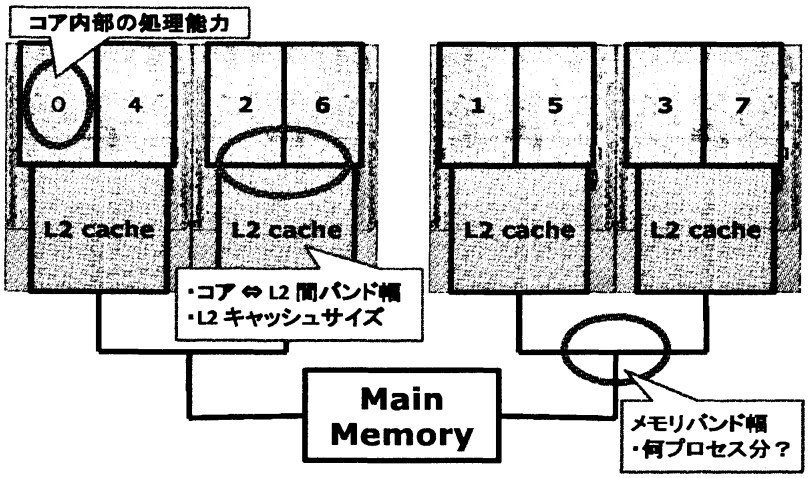

図 2: マルチコア CPU におけるボトルネックの所在

これらのクラスタ計算機で用いられている CPU は全てシングルコアなので、最近のマルチコア搭載の CPU を用いたクラスタ計算機での実験結果も見てみよう. 表 6 は各クラスタ計算機の構成や性能を示している. SDPA クラスタとは 2008 年に中央大学に設置されたクラスタ計算機で CPU に Intel Xeon 5460 を採用している. この CPU は 4 個のコアを持つマルチコア CPU であり、1 ノード内に 2 個の CPU を持っているので、図 2 のような CPU とメモリの配置、構成になっている (2CPU, 8 コア). 表 5 はこの SDPA クラスタ上で SDPARA を用いて、表 4 と同様に量子化学の問題を解いた結果である. 表 4 と表 5 を比較すると後者の方が数倍高速化されていることがわかる. しかし, SDPA クラスタにおいて 64 台と 128 台を使用したときを比較すると 128 台使用したときの方が反対に遅くなっている. この SDPA クラスタは 16 ノードなので, 128 台とは 1 ノードで 8 コアを使用することを意味する (16 ノード × 8 コア). そのため、図 2 のように、CPU コア $\Longleftrightarrow$ L2(二次キャッシュ) のバンド幅、あるいはメモリバンド幅などがボトルネックになっていると考えられる. 現在では CPU のコア数が増えているので CPU コアの処理能力だけが注目されがちであるが, それ以外にも様々なボトルネックが存在することを意識した上でバランス良く計算機の能力を使い切るソフトウェアの開発が望まれている.

4 最適化問題用オンライン・ソルバーの概要

近年、大規模かつ複雑な最適化問題を高速に解く需要は様々な産業界や学術分野において急速に高まりつつある。通常は最初に実際問題を最適化問題にモデル化 (定式化など) して、数値データを用意し最適化ソルバー (最適化問題を解くためのソフトウェア) を利用して解を求めることになる。しかし、多くの関係者の意見を総合すると以下のような問題に直面することが多く、そのことが最適化ソルバーの普及の妨げの要因になっている。

1. ユーザは必ずしもコンピュータ分野に詳しいとは限らないので、実行環境の構築 (OS やコンパイラ) やインストール作業を適切に行うことが困難な場合も多い。近年ではクラスタやグリッドなどの並列計算技術が普及し、最適化ソルバーでも並列計算に対応しているものも増えてきているが、専門家以外には並列計算の敷居が高く費用もかかることが多い。

2. ソフトウェアの更新作業等が面倒なので、最新の高性能かつ安定した最適化ソルバーが必ずしも普及していない。

3. 一つの最適化問題に対しても問題の特性や規模に対応した複数の最適化ソルバーが存在し、またパラメータ設定によって性能向上が期待できる。しかし、この選択肢の多さは専門家以外のユーザに取ってはかえって負担になることが多い。

4. 検索エンジンなどで最適化ソルバーの情報は入手できるが、どのような最適化問題に対してどのような最適化ソルバーが存在しているのか、あるいは自分が想定している最適化問題はどの最適化ソルバーで解くことができるのか等の情報収集と判断は、これもまた専門家以外のユーザには困難である。

すでに 1995 年から半正定値計画問題 (SDP) に対する主双対内点法のソフトウェア SDPA[7] 群の公開を行っている. さらに 2005 年から SDPA Online Solver [5] という Web サービスを提供している. SDPA Online Solver の特徴は以下のようになる.

[5] http://sdpa.indsys.chuo-u.ac.jp/portal/

1. Windows や Linux などから Firefox や IE 等の Web ブラウザを用いて SDPA Online Solver にアクセスするだけで SDPA[7], SDPARA[7] や SDPARA-C[7] のソフトウェアを利用することができる (図 3).

2. ユーザはパラメータファイル, データファイル, 実行するソフトウェアと計算を行うクラスタ計算機, 同時実行する CPU 数などを設定することができる.

3. ジョブマネージャーシステムなどを用いることにより, 同時に複数の並列計算要求が来たときにも対応可能である.

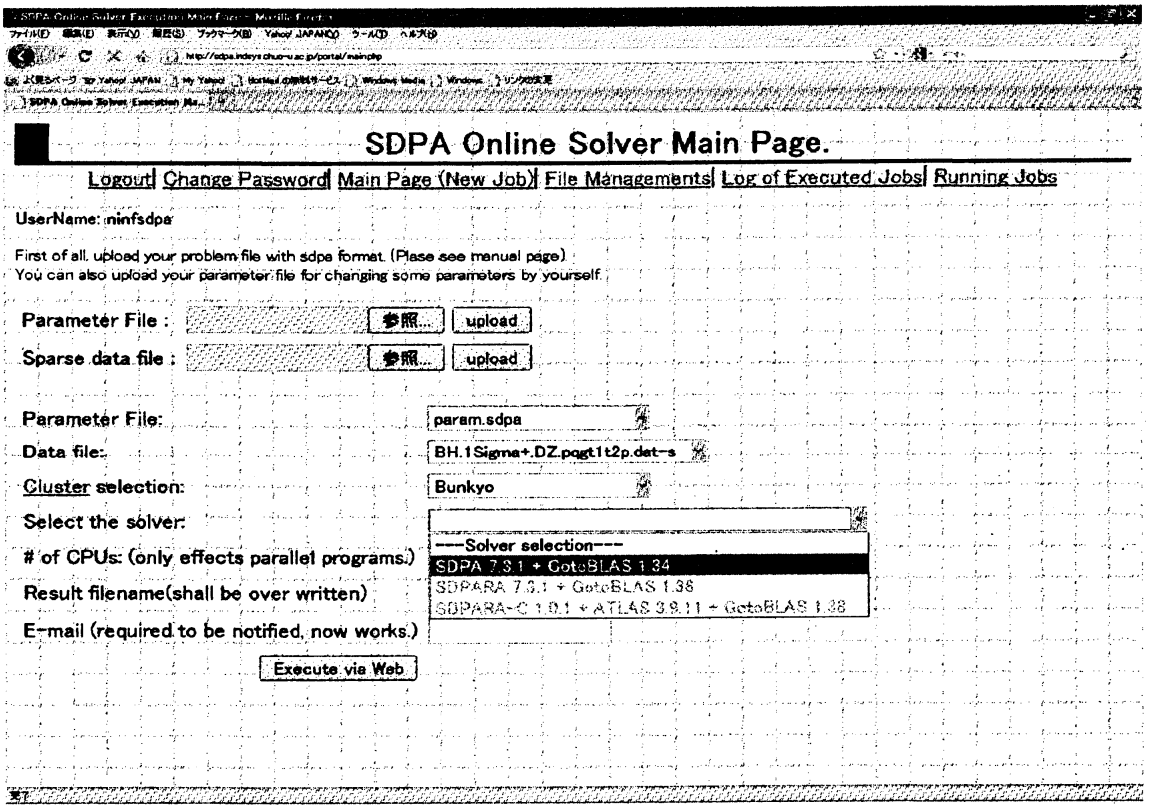

図 3: ソフトウェアの選択画面

SDPA Online Solver の Web アプリケーションはユーザーに特殊なアプリケーションを要求することなく, またプラットフォームに依存しないクライアント環境で利用可能な 3 層クライアント・サーバシステムなっている. ユーザーは, Web ブラウザさえあれば, 地理的に分散した計算リソースやクラウド技術の複雑さを気にすることなく最適化ソフトウェアを利用できる画期的なシステムになった. 図 4 は本システムの概念図である.

また, 同様のシステムで最短路問題用のオンライン・ソルバー [6]も構築, 公開を行っている. さらに今後は制約充足問題や長方形詰込み問題などの組合せ最適化問題を追加していく予定になっている.

謝辞：共同研究や情報、資料提供などをしていただきましたSDPA プロジェクトのメンバー [7], 特に東京工業大学の山下先生, また NEC システムプラットホーム研究所の高宮さん、テキサス州立大学オースティン校の後藤先生らには深く感謝致します

[6]http://opt.indsys.chuo-u.ac.jp/portal/
[7]http://sdpa.indsys.chuo-u.ac.jp/sdpa/contact.html

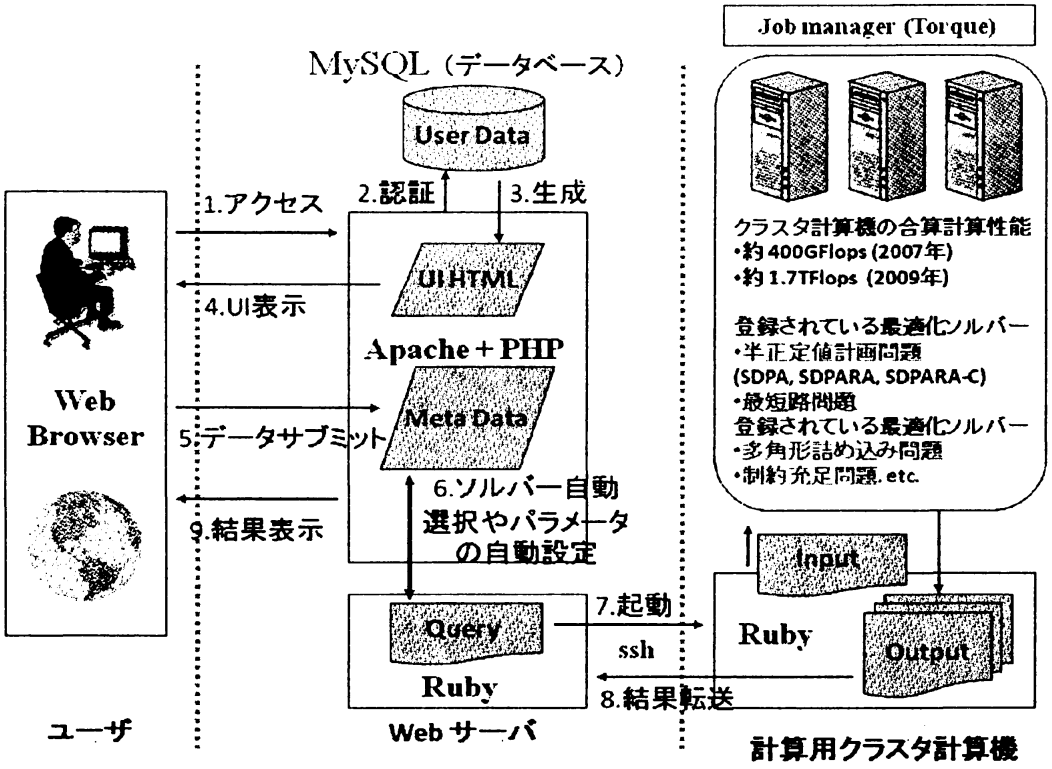

図 4: システムの概要

参考文献

[1] A.J. Coleman, "Structure of fermion density matrices," Rev. Mod. Phys. 35, 668–687, (1963).

[2] R.M. Erdahl, "Representability," Int. J. Quantum Chem. 13, 697–718, (1978).

[3] M. Fukuda, B.J. Braams, M. Nakata, M.L. Overton, J.K. Percus, M. Yamashita and Z. Zhao, "Large-scale semidefinite programs in electronic structure calculation," Research Report B-413, Dept. Mathematical and Computing Sciences, Tokyo Institute of Technology, February 2005.

[4] C. Garrod and J.K. Percus, "Reduction of the N-particle variational problem," J. Math. Phys. 5, 1756–1776, (1964).

[5] K. Goto, R. van de Geijin: Anatomy of High-Performance Matrix Multiplication. to appear in *ACM Transactions on Mathematical Software.*

[6] K. Fujisawa, M. Kojima and K. Nakata: Exploiting sparsity in primal-dual interior-point methods for semidefinite programming. *Mathematical Programming*, **79** (1997), 235–253 .

[7] K. Fujisawa, K. Nakata, M. Yamashita and M. Fukuda: SDPA Project : Solving Large-scale Semidefinite Programs. *Journal of the Operations Research Society of Japan*, **50(4)**, (2007), 278-298.

[8] 藤澤克樹, 梅谷俊治, 応用に役立つ 50 の最適化問題, 朝倉書店, (2009).

[9] M. Nakata, H. Nakatsuji, M. Ehara, M. Fukuda, K. Nakata and K. Fujisawa: Variational calculations of fermion second-order deduced density matrices by semidefinite programming algorithm. *Journal of Chemical Physics*, **114**, 8282–8292, (2001).

[10] M. Yamashita, K. Fujisawa and M. Kojima, "SDPARA : SemiDefinite Programming Algorithm PARAllel Version", *Journal of Parallel Computing*, Vol 29/8, 1053–1067, (2003).

[11] M. Yamashita, K. Fujisawa and M. Kojima, "Implementation and Evaluation of SDPA 6.0 (SemiDefinite Programming Algorithm 6.0)", *Journal of Optimization Methods and Software*, Vol 18(4), 491–505, (2003).

[12] Z. Zhao, B.J. Braams, M. Fukuda, M.L. Overton and J.K. Percus, "The reduced density matrix method for electronic structure calculations and the role of three-index representability conditions," J. Chem. Phys. 120, 2095–2104, (2004).