最適化と計算の今後
—大規模問題をどこまで解決できるのか？—

藤澤　克樹，品野　勇治

近年，大規模かつ複雑な最適化問題を高速に解く需要はさまざまな産業界や学術分野において急速に高まりつつある．これからの研究においては最先端理論 (Theory)+超大規模実データ (Practice)+最新計算技術 (Computation) の三つを有機的に組み合わせることによって，実用に耐えうる解決策の提示と大規模最適化問題を扱う際の先例となることが求められている．本稿では最適化と計算に関する最新の傾向に触れるとともに，最適化の計算の今後についても考えていきたい．

キーワード：最適化問題，高性能計算，半正定値計画問題，混合整数計画問題，グラフ解析，スーパーコンピュータ

1. はじめに

近年，最適化問題に対する研究の対象は数学的な理論から商用のソフトウェアを用いた多くの実問題の解決，さらにスーパーコンピュータ（スパコン）による大規模計算まで多岐に及んでいる．また大規模かつ複雑な最適化問題を高速に解く需要は産業界や学術分野において急速に高まりつつある．最適化問題に対するアルゴリズムおよびコンピュータに関する研究は第二次世界大戦後から発展し，今日までの約 60 年間は最適化問題に対するアルゴリズムとソフトウェアが連動しながら目覚しい発展を遂げてきた．しかし，初期のころは実社会から要求されるレベルに対して，アルゴリズムや計算機の能力が低かったため，実用に耐えうる複雑かつ大規模な最適化問題を扱うことができるようになったのは最近（21 世紀以降）のことである．

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歩しており，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つつある．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，近年ではスパコンなど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まで新世代の実装方式が開発されている．そのため，これからの研究においては

最先端理論 (Theory)+超大規模実データ (Practice)+最新計算技術 (Computation)

の三つを有機的に組み合わせることによって，実用に耐えうる解決策の提示と大規模最適化問題を扱う際の先例となることが求められている（図 1）．

本稿では最適化と計算に関する最新の傾向に触れるとともに，今後の動向や予測についても考えていきたい．

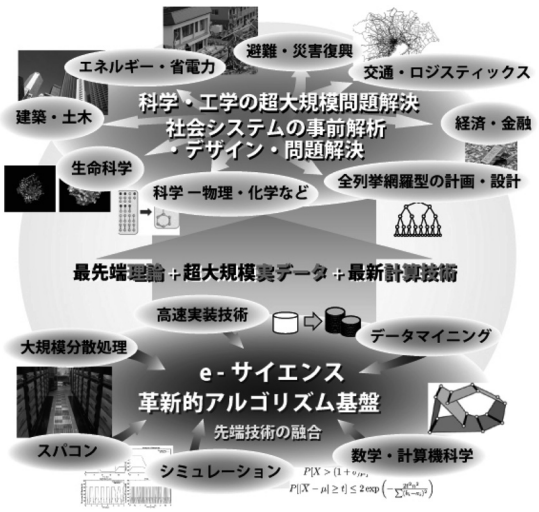

図 1　最先端理論 (Theory)+超大規模実データ (Practice)+最新計算技術 (Computation) による超大規模最適化問題の解決

ふじさわ　かつき
中央大学理工学部経営システム工学科
〒 123–0865 東京都文京区春日 1–13–27
しなの　ゆうじ
Zuse Institute Berlin, Department Optimization
Takustr. 7, D-14195 Berlin-Dahlem, Germany

2. 最適化と計算に関する最新の傾向

実社会で要求される大規模最適化問題を解決するためには，短時間に膨大な計算量とデータ量を処理するための新技術が必要となる．例えば近年の日本は地震，津波，台風，洪水など大規模災害に何回も襲われており，大規模災害時には防災計画の策定，災害時の避難と誘導および情報収集と解析，復興計画の策定，スマートグリッドによる高度かつ安定な電力供給などを行う必要があると言われている．このように発生後に早急な解決が望まれる現実問題においては，計算量やデータ量などの規模が大きく従来の手法では処理が困難であり，大規模ネットワークの探索とグラフクラスタリングなどの高速処理技術の開発が必要とされている．現在では大規模な計算基盤としてペタスケールスパコンが用いられている．2015 年頃にポストペタスケールスパコン，さらに 2018 年から 2020 年頃にエクサスケールスパコンの登場を目指して，日本，米国，欧州，中国などで研究開発が行われている．しかし，現在の最適化理論とソフトウェア実装方法では数千万規模の並列性を備えているポストペタスケールシステム上でのスケーラブルなソフトウェアの並列実行は困難であり，アルゴリズムとシステムソフトウェアの同時並行的な解決が求められている．そのため理論的性能限界などからボトルネック箇所を特定，数値演算能力とメモリバンドなどのトレードオフ関係を把握，計算量とデータ移動量の正確な推定，疎性やサイズなどのデータ特性と性能値の見極めなどに関する研究開発の必要性が高まりつつある．

近年，最適化および関連分野では大きな変化が起きているが，ここで簡単に最適化問題に関する最新の傾向について触れてみたい．

1. 計算量による理論的解析の優劣とソフトウェア実装後に計算機で実行したときの計算結果が大きく異なるという現象が見られる．その理由としては以下のような原因が考えられる．
   (a) 理論的な解析時に想定されている最悪な場合という状態が計算機上で扱うことのできる問題の範囲では発生しにくい．例えば，現在の計算機においては整数型の変数は大きくても $2^{64}-1$ であるので，計算機で扱うことができる範囲では $\log_2 n$ 関数は大きくても 64 程度である（つまり定数とみなすこともできる）．
   (b) 計算量の評価において，演算量とデータ移動量が混同されている（一般的には後者のほうがコストが高い）．
   (c) 単なる性能値ではなく，電力あたりの性能値も重要となっている（図 2）．1 個のデータ移動に必要な電力は，1 個のデータの演算に必要な電力の 100 倍になることもあり，今後は高速性と省電力性を両立させたアルゴリズムの開発が必要．
2. 最適化ソフトウェアと最新の計算技術の密な融合によって，後述するように一部の最適化問題では驚くほど大規模な問題が高速に処理できるようになった．
   (a) 混合整数計画問題 (Mixed Integer Program; MIP)：Gurobi や CPLEX などの商用ソルバが有名だが，研究機関においても SCIP[1] や CBC[2] などのソルバの開発も行われている．MIP ソルバの進歩に関しては，第 5 章参照．
   (b) 巡回セールスマン問題 (Traveling Salseman Problem; TSP)：数万から 10 万点規模の大規模問題が扱われている[3]．
   (c) 半正定値計画問題 (Semidefinite Program; SDP)：著者らのグループではソフトウェアを用いて SDP に対する大規模，高速かつ安定計算を行っている [1, 2]．2013 年 9 月には東京工業大学スーパーコンピュータ TSUBAME2.5 を用いて 233 万制約を持つ SDP を解くことに成功している．

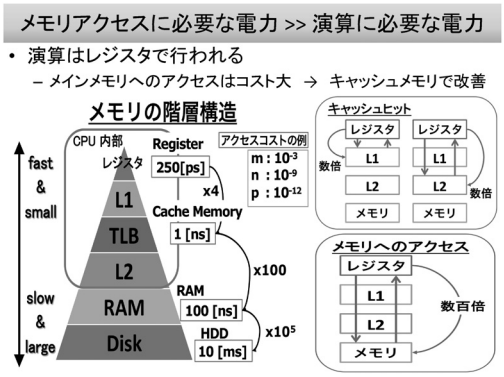

図 2 データ移動のコストと消費電力

[1] http://scip.zib.de
[2] https://projects.coin-or.org/Cbc
[3] http://www.tsp.gatech.edu/

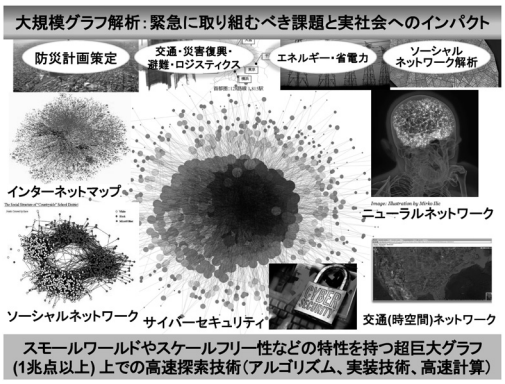

図 3　大規模グラフ解析とその応用

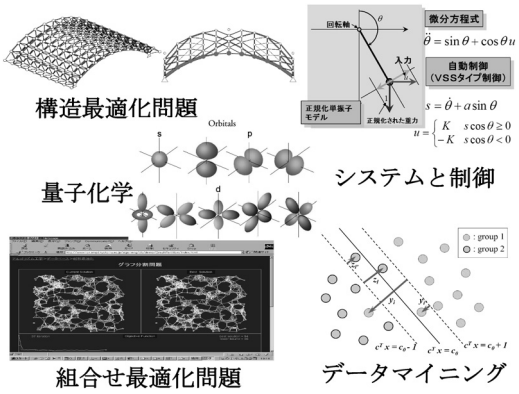

図 4　SDP に関する主要な応用分野

3. 最適化問題の巨大化によって倍精度演算（仮数部 52bit）の精度不足の問題が顕著になり，4 倍精度や多倍長演算などの高精度計算が用いられるようになった．

3. 大規模グラフ解析の応用と今後の計画

この節では基本アルゴリズムとして近年注目度の高いグラフ解析（幅優先探索，最短路計算，コミュニティ抽出など）を取り上げる．グラフ解析に関しては数年以内に数千万規模の並列性を備えたポストペタスパコン上での高性能な超大規模グラフ処理技術が確立され，ストレージの階層性が深化したポストペタスパコン上での高性能な超大規模グラフ処理技術が開発されていくと予想されている．これによって一義的には，防災計画の策定，災害時の避難と誘導および情報収集と解析，スマートグリッドによる高度かつ安定な電力供給など，安全安心な社会基盤実現に貢献することが可能となる．以下の分野に対してグラフ解析を適用することを想定している（図 3）．

1. 交通データに対する経路探索：動的に変化する交通量等から高速な重要度判定を行うことによって，交通管制などに活用する．
2. ソーシャルネットワークデータ（マイクロブログや SNS など）やウェブデータに対する動的な重要度，影響度の判定．各点の周辺，および広域内における影響（情報の伝播力）を推定する．
3. その他：疫病の拡散，人口の増減，経済動向等の分析．ライフライン等の基盤計画（電力，水，食料）．生命科学系（創薬，遺伝子）．ビジネス系（金融，データマイニング）．安全保障分野（組織構成の解明，事件事故の事前予測）．

4. 半正定値計画問題ソルバの現状と可能性

この節では注目度の高い最適化問題の中から，近年の研究進展が著しく 21 世紀の線形計画問題として幅広い応用を期待されている半正定値計画問題（SDP）を取り上げる [3, 4]．SDP などの最適化問題が最近特に注目を集めている理由には以下のようなものが考えられる．

1. **高速で安定したアルゴリズムの存在**：主双対内点法などのアルゴリズムによって多項式時間で最適解を求めることができる．
2. **広範囲の問題に適用可能**：SDP は線形計画問題 (Linear Programming Problem; LP)，凸二次計画問題 (Convex Quadratic Programming Problem; CQP) や二次錐計画問題 (Second-Order Cone Programming Problem; SOCP) などを含んだ凸最適化問題の枠組であるが，非凸最適化問題に対する強力な緩和値を導き出すこともできる．そのため SDP を繰り返して解くことによって，最適解を求めることが極めて難しいが，実用上重要な非凸最適化問題（例えば双線形行列不等式（Bilinear Matrix Inequality; BMI など）を扱える可能性を持っている．
3. **多彩な応用分野**：組合せ最適化問題，整数計画問題，ノルムなどを用いた配置問題，システムと制御，ロバスト最適化，量子化学など非常に多くの応用が存在する（図 4）．
4. **公開されているソフトウェア利用により大規模問題への対応も可能**：多くのソフトウェアが開発され，インターネットより公開されている（ [2] など）．さらに複雑で大規模な問題を解くためには，理論的成果を随時組み入れるとともに，最

表 1 東工大スパコン TSUBAME 2.0 および 2.5 での大規模計算時における SDP の制約条件数

問題名（二次割当問題）	制約条件数
QAP6	709,275
QAP7	1,218,400
QAP8	1,484,406
QAP9	1,962,225
QAP10	2,339,331

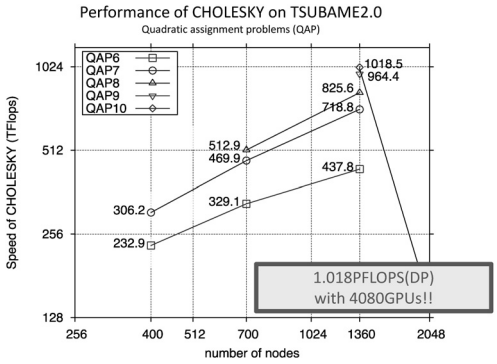

図 5 東工大スパコン TSUBAME2.0 および 2.5 での SDP の大規模計算時における Cholesky 分解の性能：org は SDPARA 7.5.0 (2012)，new は SDPARA 7.6.0 (2013) を示す．各問題の制約条件数については表 1 を参照．

新の並列計算技術（クラスタ&グリッド&クラウド計算）などとの融合も必要不可欠である．最近では後述するように非常に大きな規模の SDP を解くことが可能になってきている．

すでに述べたように SDP に対しては高速かつ安定した反復解法である内点法アルゴリズムが存在しているが，巨大な線形方程式系の生成と計算が大きなボトルネックとなっている．著者らのグループでは，疎性の追求，計算量やデータ移動量などによる計算方法の自動選択などの技術をほかに先駆けて実現して，世界最大規模の SDP を高速に解くことに成功している．具体的には主要なボトルネックの 1 つである線形方程式系の Cholesky 分解に対して，多数 GPU の活用や計算と通信のオーバーラップ技術を応用することによって，制約式の数が 233 万以上となる世界最大規模の巨大 SDP を解くことに成功した．このとき 4,080 個の GPU を同時に用いて最大で 1.713PFlops の性能を達成した（図 5）．世界的に見ても数学系ソフトウェアでペタフロップスを超える性能を出す例はほとんど見られない．今後も大規模な SDP を解く研究が推進されていくことによって，2020 年頃のエクサスケールスパコンの登場時には最大で 1,000 万制約を持つ問題を解くことも可能になると推定されている．

5. 汎用混合整数計画問題ソルバの現状と可能性

混合整数計画問題 (MIP) は，現在，最も使われている最適化問題であると思われる．Robert. E. Bixby 博士が，CPLEX の開発グループから離れて Gurobi 開発を開始した後，Zuse Institute Berlin におけるセミナーにおいて（2009 年），80%以上の CPLEX ユーザの対象としている問題は，（混合）整数計画問題だと話している．

大規模グラフ解析や半正定値計画問題ソルバの基本的なアルゴリズムの計算量が多項式時間であり比較的正確な将来の予測ができるのに対して，混合整数計画問題は NP 困難な問題であるため，本質的に予測が困難である．本節では，汎用混合整数計画問題ソルバ（MIP ソルバ）に限定し，理論面の話ではなく，実際に計算するためのプログラム開発という立場から，その現状と今後について可能性を考える．また，モデリング言語内で問題の構造を抽出して解いたり，Decompositon 技術を利用せずに，MIP のインスタンスを単に MIP ソルバで解くという場合に限定する．MIP ソルバでは，アルゴリズム改善による性能向上のほうが，計算機アーキテクチャの進歩による性能向上をはるかに超えることが知られている（報告例，[5] 参照）．NP 困難な問題を扱う際には，線形の性能向上では太刀打ちできないので，解けるインスタンスを増やすためには，基本的には何らかのアルゴリズム上の工夫が必要となる．実装を考えると，そのようなアルゴリズム上の工夫を追加することは，プログラムの規模が大きくなることであり，結果として，これまで解けていた問題を解く時間が長くなる可能性はあり，実際にそのようなケースもある．しかし，最新の CPLEX に関する論文 [6] でも，汎用 MIP ソルバとしての一般的な性能向上は著しく，かつ，MIP ソルバのユーザがそれを実感できているはずである．[6] に示されているように，背景にあるのは注意深く設計された大量の数値実験を通しての地道な開発と徹底的なチューニングである．

最先端のソルバに対しては，第 2 節で述べられているような，計算機アーキテクチャを意識した開発が行われている．単一コアによる速度向上が困難になり，マルチ・コアによる速度向上を考える必要がある状況に対応して，Gurobi は，共有メモリ環境上で動作する並列処理の利用を最初から前提としたソルバとして開

発された．特に，近年の MIP ソルバは並列処理の利用を強く意識している．また，NUMA (Non-uniform Memory Acess) での性能低下の原因を確認する努力を行っている．単に，優先順位としてアルゴリズム改善への比重が未だに大きいのが現状である．ここでは，まずその背景について公開されアクセス可能な MIPLIB での統計と CPLEX の性能測定に関する論文内容に基づき考察する．その後，著者の MIP ソルバ開発グループでの経験に基づき，今後の MIP ソルバの可能性に関する私見を述べる．

5.1 MIPLIB

NP 困難である最適化問題を扱う際には，変数の数，制約式の数が同じであっても，解くインスタンスによってソルバの性能は大きく変わる．したがって，ソルバの性能を共通のインスタンスセットによって評価することは極めて重要である．現実問題を MIP として定式化した際のインスタンスでの性能を計測したい．この場合，何らかのプログラムによりランダムに生成されるインスタンスは，極端に簡単か，極端に困難かのどちらかになりがちで，実際の利用に際して高性能であることを実感できるソルバを効率的に開発するためには適当でない．現在，最先端の MIP ソルバ開発を行っている研究者は，この点を強く意識しており，極めて慎重にベンチマークとなるインスタンスセットを集めている．最初の MIPLIB から 5 版目になる MIPLIB2010[5] は，商用・非商用の主要高性能 MIP ソルバ開発者全員が 2009 年に会議を行い，どのような目的で，どのようにインスタンスを収集し，どのようにベンチマークとなるインスタンスセットを選択するかに関して議論し，1 年間をかけて世界中からインスタンスを集め，すべての主要 MIP ソルバでのパラメタ設定検討のため何度も解き，ベンチマークセットが作られている（詳細は [5] 参考）．

ここでは，2013 年 9 月 18 日における MIPLIB2010 の全インスタンスに対して，easy（通常の PC 上でインスタンスを 1 時間以内に解く MIP ソルバがある場合に easy と分類される），hard，open（最適解が知られていないインスタンス，本稿においては “not solved” とも記す）と分類されているインスタンスを，その規模を 0-1 変数を含む整数変数の数，および，制約式の数によって規模を示す平面上にプロットする．図 6 は，全変数が整数変数のインスタンスである．図 7 は連続変数を含むインスタンスである．左端が整数変数であり，プロットの際に右方向へ連続変数分を線により示すことで，全変数の数も同時に示している．MIPLIB2010

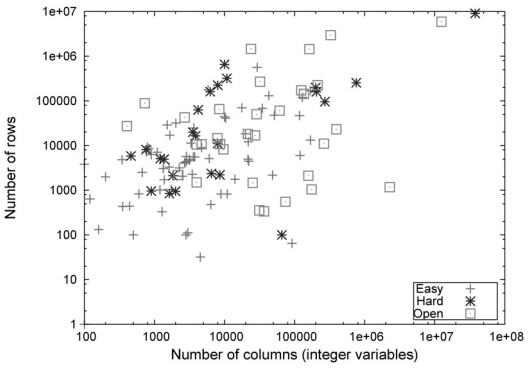

図 6 MIPLIB2010 の IP/BP の変数・制約式の数と難易度

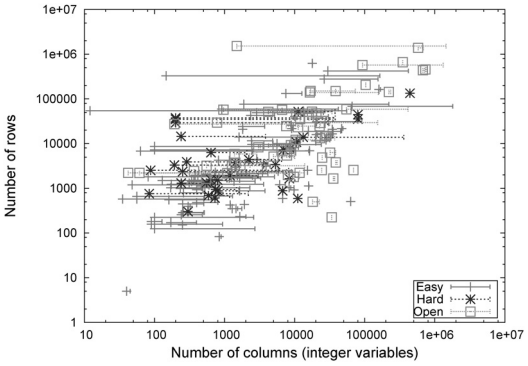

図 7 MIPLIB2010 の MIP/MBP の変数・制約式の数と難易度

のインスタンスが慎重に選ばれていることを考慮すると，図 6，7 から，問題の難易度を制約式の数や，変数の数だけで議論することが大きな誤りであることが確認できる．実際，MIPLIB2010 の全インスタンスにおいて，変数の数，制約式の数での最大規模の問題はすでに解かれている（実行不可能解[4]）．現在の MIP ソルバは，内部的には一度前処理により再定式化した問題を解いている．したがって，このようなプロットそのものが実際にソルバ内部で解いている問題の規模に対応していない．MIPLIB2010 の benchmark set の全インスタンスにおける SCIP ver.2.1 による前処理後のインスタンスの規模は [8] に掲載されている．

5.2 ソルバ・フレームワーク

現在の最先端 MIP ソルバは，いずれもソルバ・フレームワークと考えられる大規模ソフトウエア・システムとみなしたほうがよい．全体の動作は LP ベースの分枝限定法であるが，前処理 (Presolving)，分枝変数選択 (Branching)，切除平面生成 (Cutting planes)，暫

[4] 実行不可能性は，変数や制約式が多くても前処理などにより検出されることがある．

定解生成ヒューリスティック (Primal heuristics)，などの基本構成要素機能がプログラムとして追加・置き換え可能な構造になっており，それぞれの要素機能の拡張が容易である．CBC[5]，SCIP は，ソース・コードが公開されているので，すべてを確認することができる．本原稿執筆時点での SCIP のバージョン 3.0.1 の C ソースコードはすでに 500,000 行に達しているが，それぞれの機能の拡充が日々なされており規模は拡大し続けている．特に，近年におけるソルバ・フレームワーク全体に対して作用するような大きな変更は，おそらく，CPLEX の dynamic search であると思われる．企業秘密であり何を行っているかはわからないので曖昧な表現しかできないが，参考文献 [6] には，dynamic search は，特定の機能の追加ではなく，各機能間の相互作用を実行時に収集している情報に基づき動的に変更していると書かれている．

5.3 再現可能性

各要素機能に対して新たなアルゴリズムが追加される際には，そのインパクトを計測する必要がある．正確にインパクトを知るためには数値実験を通して反証可能 (falsifiable) であることが大前提なので，ソルバ開発者はプログラム実行の再現可能性（決定性）に強くこだわっている．「プログラムが逐次処理だから決定的に動作する」とは言えない．汎用の MIP ソルバであれば，前処理やカット生成の停止条件として，「実時間」を使いたいと考えるのは，むしろ自然であり，実時間を使った場合，逐次処理でもタイミングにより非決定的な振る舞いとなる．現在の最新 MIP ソルバは並列探索の機能を持っているが，商用ソルバはすべて決定性並列がデフォルト設定である．つまり，同じ動作環境，同じ数のスレッドを利用する場合の結果が同じであることを保証するように動作する．

5.4 性能変動

性能変動 (performance variablity) は，MIP ソルバの性能に関する技術用語である．MIP ソルバの分枝変数選択，カット生成など各機能内では，複数の候補に対して評価値計算を行い，候補の中から評価値の高いものを選択して計算が進むという処理がほうぼうで行われる．入力データの行や列の並びを変えるだけで，同じ評価値の別の要素が選択されることになり，性能が大幅に変わることがある [5]．実行ファイルを生成する際のコンパイラが変わる，あるいは，実行時の CPU のアーキテクチャが変わることなどでも，同様のことが起こる．このような現象を性能変動と呼ぶ．MIP ソルバの現状では，実行時の計算機アーキテクチャが変わることによる性能変動よりも，アルゴリズム上の選択基準が変わることによる性能変動のほうが大きい．

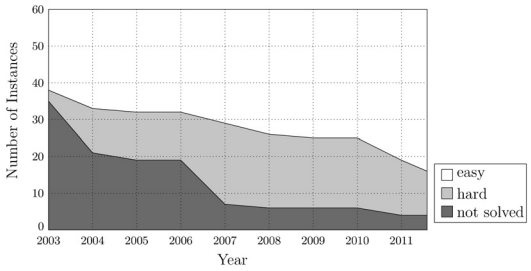

図 8　MIPLIB2003 の easy，hard，not solved の推移

単なる幸運によりインスタンスが解けたという結果を排除したソルバの性能を計測したいので，性能変動は，MIP ソルバの性能測定を極めて困難なものとする．CPLEX では，乱数の種を変えることで，最初の LP 緩和問題に対する異なる最適基底解の選択や，暫定解を求めるヒューリスティックで必要とされるさまざまな選択基準が変わることになる．[6] では，乱数の種を変えた実験結果に基づく，性能変動の効果を考慮した評価方法を示してる．

5.5 MIP ソルバの進歩と現在の MIP ソルバにおける要素機能のインパクト

MIP ソルバの進歩に関しては，CPLEX のバージョンが上がるにつれ，どのように性能が向上したかについて，[6, 7] に詳細に述べられている．特に，[6] では，CPLEX 12.5 に関して，要素機能のインパクトを詳細に調べている．また，[7] の結果と矛盾する部分に関しては，それが何に起因するかも調べており，要素機能のインパクトを正確に計測するための数値実験結果の比較方法，および，インスタンスの選択方法を含む，インパクト計測手法を開発しているともみなせる．要素機能のインパクトは，要素機能を一つずつ外した数値実験により計測されている．CPLEX 12.5 では，前処理のインパクトが最も大きく，次に分枝変数選択のインパクトが続く．カット生成のインパクトは 3 番目であり，これが [7] と異なる．カット生成のインパクトが，[7] において大きかった原因は，数値実験に利用したインスタンスセットの偏向に起因している．このように，インスタンスセットを抜きにした性能の議論は，MIP ソルバでは困難なため，本稿では MIPLIB に限定している．

[5] 開発者の John Forrest が引退後，主たる後継者が見つかっていないことから，本文で述べている要素機能の拡張は容易ではないと予想される．

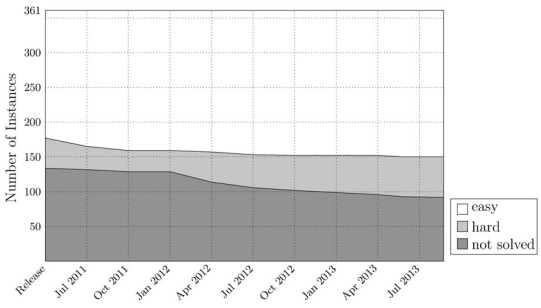

図 9　MIPLIB2010 の easy，hard，not solved の推移

図 8 は，MIPLIB2003 の easy，hard，open (not solved) に分類されるインスタンス数がどのように推移してきたかを示している．最後に最適解が与えられた 2 問，stp3d，ds は，ParaSCIP によりスーパーコンピュータ HLRN II 上で解かれた [11]．現在，残っている，dano3mip，liu，momentum3，t1717 の 4 問は，常にさまざまな試みがなされているにもかかわらず，未だに open である．

5.6　今後の可能性

各 MIP ソルバのバージョンが上がるごとに，MIPLIB2010 の open インスタンスの数は確実に減るはずである．バージョンアップにより，open に属するインスタンスが解かれると同時に，hard に属するインスタンスが easy へと移行する傾向にある．つまり，簡単に解けるか，あるいは，全く歯が立たないかのどちらかに多くのインスタンスが分類されてゆく．図 8 は，それを示唆している．実際，図 9 の現状においても hard に属するインスタンスは少ない．また，[9] でもこの傾向が近年特に強まっていることが指摘されている．そこで，全く歯が立たないというインスタンスのどれだけが解けるようになるかが勝負である．スパコンのように多数のコアを使える状況での予測は，[10] にまとめられている．本稿では，[10] の内容とは若干異なるかもしれないが，著者の私見を述べたい．

分散メモリ環境上での MIP ソルバという観点では，ParaSCIP[11] が数年先行している．商用の MIP ソルバ開発者は，その動きを理解しておりそれぞれの商用ソルバに機能の一部は組み込まれ始めている．ParaSCIP は，MIP ソルバを並列化するソフトウェア・フレームワークである，Ubiquity Generator Framework[12] 上に実装されており，CPLEX を MIP ソルバとして利用して ParaCPLEX も構築でき実際に動作している．この場合，CPLEX の強力な MIP ソルバとしての性能を大規模並列処理に適用できることになると考えられる．しかし，残念ながら CPLEX をライブラリ利用して，コールバックルーチンを追加すると，まず，dynamic search は利用できなくなり，さらにさまざまな要素機能の性能が下がる[6]ため，CPLEX 本来の性能を超えることは容易ではない[7]．CPLEX 12.5 にはすでに，remote object により分散メモリ並列処理環境を意識した機能が追加されている．今後は，CPLEX 本来の機能して，ParaCPLEX が行っていることを remote object により実現されていくと予想される．CPLEX が分散メモリ環境で動作する並列 MIP ソルバとなっても，少なくとも限界を確認するまでは決定性並列による実装を試みるはずである．なぜなら，MIP ソルバの場合には，CPU コアの利用率が問題を解くという観点での効率に対応しないため，[6] に示されているような効果確認を繰り返しながら改善されてゆくはずである．仮に，意識的に CPU コアを遊休状態にしてでも，解法全体をコントロールするほうが，問題を解くうえでは効率的である．どうしても歯が立たない問題に対しては，近い将来分散メモリ環境上で動作する並列 MIP ソルバの利用が主流になると予想される．

そこで，現在の ParaSCIP がどの程度の規模の分散メモリ環境までスケールアップできるかが知りたい．著者が実施した最大規模のものは，オークリッジ国立研究所の Titan を利用して行った，35,200 コアでの並列実行である（具体的には dano3mip を解くことを試みた）[13]．極めて困難なインスタンスを解く際には，全体を制御するプロセスが 1 つしかない ParaSCIP は，この規模で 12 時間の制限時間まで安定して動作した．決定性並列を実現した場合には，明らかにこの規模での動作は期待できないが，決定性を外せればこの規模までは確実に動くと言える[8]．

Ubiquity Generator Framework の設計方針は，部分問題を解く際に，高性能 MIP ソルバをそのまま利用することである．つまり，分散メモリ環境上であっても，MIP ソルバは 1 計算ノード内で動作する．この設計方針は，今後の分散メモリ環境上で動作する並列 MIP ソルバにおいても引き継がれる可能性が高い．

---

[6] このため，CPLEX をカスタマイズして，CPLEX との性能比較をしている論文では，何も行わないダミーのコールバックを追加した CPLEX を比較対象としていることが多い．手法の確認には適当であるが，本来の CPLEX の性能を超えることはほとんどない．

[7] それでも，ParaSCIP と比較すると ParaCPLEX の性能は格段に良くなる．

[8] 現在の ParaSCIP には決定性並列が実装されている．ただし，デバッグのための実装であり，性能は逐次処理プログラムよりも遅い．実現方法に関しては [12] 参照．

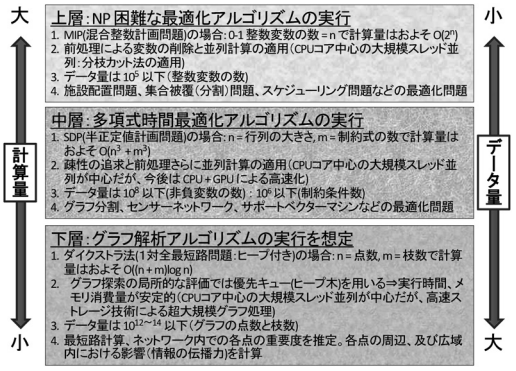

図 10　今後想定される最適化問題：計算量と問題規模

（前処理後の）変数の数，制約式の数という観点で規模を議論するなら，ここがボトルネックになる．つまり，1 計算ノードのメモリで動作する必要があり，緩和問題である LP が解けるかどうかが鍵になる．よって，この点におけるブレークスルーは，分散メモリ環境上で効率的に動作する線形計画ソルバが実現できるかどうかである．スパコンは GPU を搭載するのが常識となりつつあるので，GPU も利用して分散メモリ環境上で動作する大規模 LP が効率的に解けるようになれば，大きなブレークスルーになるはずである．残念ながら，著者の知る限り，現状では GPU を利用した LP ソルバの開発の成功例はない．

6. おわりに

今後は最適化問題の応用分野が非常に広範化（企業，社会，公共政策）するとともに，巨大なセンサーデータによる最適化問題の複雑化＆巨大化が予想される（図 10）．問題サイズを n とした場合では，計算量に応じて $O(n^3) \rightarrow$ 数百万程度，$O(n \log n) \rightarrow 100$ 億以上，$O(n) \rightarrow 100$ 兆程度の問題規模を 2015 年頃に登場する予定のポストペタスパコン上で扱うことができるようになると予想されている．われわれが推進している JST CREST プロジェクト[9]では，今後 5 年以内にスーパーコンピューター等で以下の問題に対するアルゴリズムとソフトウェアの開発を目標としている．

1. 超大規模ネットワークに対する最適化，探索およびクラスタリングアルゴリズム
   (a) 最短路（中心性），最大フロー，PageRank，グラフ分割，中心性，その他
   (b) 数理計画問題 (SDP，MIP): 高性能汎用ソルバの開発

[9] http://www.graphcrest.jp/jp/

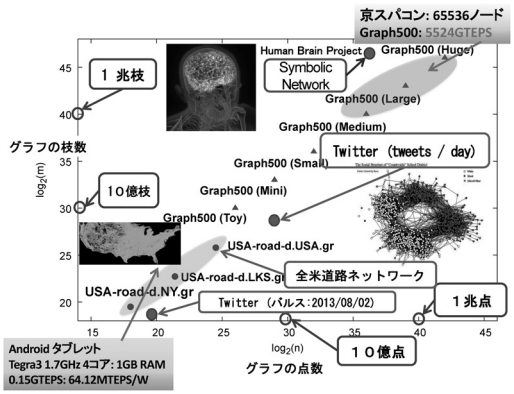

図 11　今後想定されるグラフ解析の応用と問題規模

   (c) グラフクラスタリング，高速グラフレイアウト
2. 数百万頂点～数兆頂点，数億枝～数百兆枝からなる超大規模なグラフ解析（図 3, 11）
   (a) 2^{42} 頂点　数 PB 以上のワーキングデータセット
   (b) グラフ解析：例：890 億個のニューロンとその接続 100 兆枝 ペタバイト級のストア領域：（2023 年：エクサスパコン）
   (c) 5 億頂点 250 億辺のグラフのインタラクティブな操作のための計算性能

今後もアルゴリズムやソフトウェア技術の飛躍的な進展によって，すでに述べたように極めて広範囲な分野にまで最適化問題の適用が拡大されていくことが期待できる．

謝辞　図 8，9 を作成し提供してくださった Gerald Gamrath 氏に感謝する．

参考文献

[1] K. Fujisawa, T. Endo, H. Sato, M. Yamashita, S. Matsuoka and M. Nakata, High-Performance General Solver for Extremely Large-scale Semidefinite Programming Problems, Proceedings of the 2012 ACM/IEEE Conference on Supercomputing, SC'12, 2012.
[2] K. Fujisawa, K. Nakata, M. Yamashita and M. Fukuda, SDPA Project: Solving large-scale semidefinite programs. *J. Oper. Res. Soc. Japan*, **50**, 278–298, 2007.
[3] Handbook of Semidefinite Programming (International Series in Operations Research and Management Science), ed. by H. Wolkowicz, R. Saigal and L. Vandenberghe, Kluwer Academic Pub, 2000.
[4] Handbook on Semidefinite, Conic and Polynomial Optimization (International Series in Operations Research and Management Science), ed. by M. F. Anjos

and J. B. Lasserre, Springer Science+Business Media, 2011.
[5] T. Koch, T. Achterberg, E. Andersen, O. Bastert, T. Berthold, R. E. Bixby, E. Danna, G. Gamrath, A. M. Gleixner, S. Heinz, A. Lodi, H. Mittelmann, T. Ralphs, D. Salvagnin, D. E. Steffy and K. Wolter, MIPLIB2010 Mixed Integer Programming Library version 5, *Mathematical Programming Computation*, **3**, 103–165, 2011.
[6] T. Achterberg and R. Wunderling, Mixed Integer Programming: Analyzing 12 Years of Progress, Facets of Combinatorial Optimization—Festschrift for Martin Grötschel, 431–462, Springer, 2013.
[7] R. E. Bixby and E. Rothberg, Progress in computational mixed integer programming—A look back from the other side of the tipping point, *Annals of Operations Research*, **149**, 37-41, Kluwer Academic Publishers-Plenum Publishers, 2007.
[8] 品野勇治，T. Achterberg, T. Berthold, S. Heinz, T. Koch, S. Vigerske and M. Winkler，制約整数計画ソルバの並列化，統計数理，**61**, 47–78, 2013.
[9] T. Koch, A. Martin and M. E. Pfetsch, Progress in Academic Computational Integer Programming, Facets of Combinatorial Optimization—Festschrift for Martin Grötschel, Springer, 483–506, 2013.
[10] T. Koch, T. Ralphs, and Y. Shinano, Could We Use a Millon Cores to Solve an Integer Program?, *Mathematical Methods of Operations Research*, **76**, 67–93, 2012.
[11] Y. Shinano, T. Achterberg, T. Berthold, S. Heinz and T. Koch, ParaSCIP: A Parallel Extension of SCIP, Competence in High Performance Computing 2010, Springer, 135–148, 2012.
[12] Y. Shinano, S. Heinz, S. Vigerske and M. Winkler, FiberSCIP—a shared memory parallelization of SCIP. ZIB-Report 13–55, Zuse Institute Berlin, 2013.
[13] Y. Shinano, T. Achterberg, T. Berthold, S. Heinz, T. Koch and M. Winkler, Solving hard MIPLIB2003 problems with ParaSCIP on Supercomputers: An update. ZIB-Report 13-66, Zuse Institute Berlin, 2013.