小特集 高速点火核融合の統合シミュレーション
－多階層プラズマシミュレーションシステム“FI^3”－

2．多階層統合シミュレーションシステム FI^3 の概要

坂上仁志，坂口智哉[1]，三間圀興[2]

核融合科学研究所 理論・シミュレーション研究センター，[1]兵庫県立大学大学院工学研究科

[2]大阪大学レーザーエネルギー学研究センター

（原稿受付：2005年12月2日）

高速点火核融合においては，全体的な爆縮流体運動，レーザープラズマ相互作用と高速電子生成，高密度プラズマ中の高速電子伝播および高速電子によるコア加熱が重要な研究テーマである．しかし，これらの物理現象の時間および空間スケールは大きく異なるため，すべての現象を単体のコードでシミュレーションすることはできない．そこで，それぞれの物理を別の複数のコードでシミュレーションし，コード間でデータ転送することによって全体を統合する多階層統合シミュレーションシステムを開発した．この統合におけるデータ転送には，TCP/IP をベースとした軽装な専用プロトコルを設計，実装した．今後，このシステムを使って，高速点火実験と直接比較しうる規模のシミュレーションを実施する計画である．

Keywords:

fast ignition, distributed computing, code integration, communication protocol

2.1 はじめに

従来のホットスポットによる中心点火方式とは異なって，爆縮と点火をまったく別のステップで実現する高速点火の概念が考案されている[1]．高速点火とは，比較的長いパルスのレーザーによって低温で高密度に圧縮されたターゲットに，超高強度の短パルスレーザーを打ち込んで点火を行う方式であり，中心点火方式よりも高利得が期待でき，点火に必要なドライバエネルギーの大幅な低減が期待されている．この高速点火の物理を解明するためには，ミクロな現象とマクロな現象が相互に入り組んで絡み合った複雑現象の全体的な理解が必要であり，大規模コンピュータシミュレーションは，そのための有効なツールと考えられている．高速点火において解明が必要な物理を列挙すると，1）爆縮用レーザーの吸収とオーバーオールなターゲットの輻射を含む爆縮による流体運動，2）加熱用超高強度レーザーの吸収と高速電子の生成，3）高密度プラズマ中での異常抵抗と高速電子の伝播，4）高速電子のコア中でのエネルギー緩和とコア加熱であり，これらをミクロからマクロまでシームレスな視点から詳細に研究しなければならない．しかし，これら4つの物理現象は，時間や空間のスケール，扱うべき物理量がそれぞれ大きく異なるため，1つの巨大なコードですべてをシミュレーションすることは難しい．そこで，4つの現象を別々の独立したコードでシミュレーションし，お互いがデータ交換しながら全体として協調動作し，統合的にシミュレーションできるシステムが必要となる[2,3]．統合シミュレーションシステムでは，各コードは個別に動作するため，データ交換のインターフェイスさえ決めれば，コード開発そのものは異なった組織において，コードごとに独立に行うことができる．このため，各コードの保守や機能追加は，それぞれの開発者に委ねることができ非常に容易となる．一方，コンピューティング環境に目を向けると，大規模シミュレーションには並列コンピュータの利用が不可欠であるが，ベクトル／スカラというプロセッサ構造，共有メモリ／分散メモリというメモリ構成，汎用の Ethernet／専用のインターコネクト等のプロセッサ間ネットワークなど，様々なアーキテクチャを採用した並列コンピュータが存在する．各コードを高効率かつ高速に実行するためには，コードごとの計算アルゴリズムに適したアーキテクチャを持つ並列コンピュータを，その計算粒度に適した規模のプロセッサ数で利用することが望まれる．この統合シミュレーションシステムでは，すべてのコードを同一の並列コンピュータ上で実行する必要がないので，コードごとにコードの特性に対して最適な並列コンピュータを複数のサイトの中から適材適所で選ぶことが可能になる．また，コードを並列化する手法には，HPF（High Performance Fortran）[4]，OpenMP[5]，MPI（Message Passing Interface）[6]および自動並列化などの様々な方法があるが，統合シミュレーションシステムにおけるデータ交換は専用のプロトコルを用いて行うため，並列化の手法には依存しないで考えることができ，プログラミングを簡便にしている．

以上述べたアイデアを用いて高速点火を総合的に解析できるシステムの開発を目指し，多階層統合シミュレーションシステム FI^3（Fast Ignition Integrated Interconnecting code）の研究を開始した[7]．FI^3では，まず，オーバーオールな流体運動は，物質の空間スケールと形状が大きく変化することを考慮して ALE（Arbitrary Lagrangian Eulerian）法を用いた輻射・流体コード[8]を用いる．次の超高強度

corresponding author's e-mail: sakagami.hitoshi@nifs.ac.jp

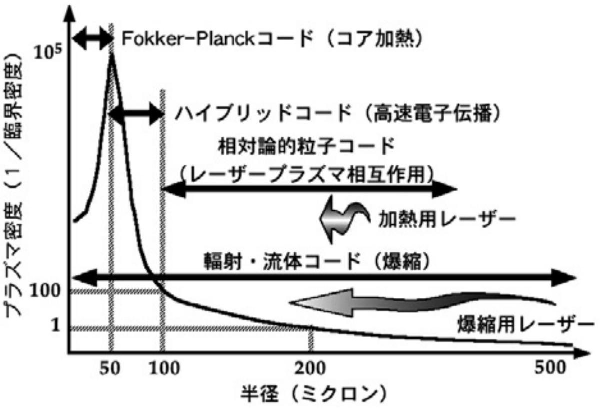

Fig. 1 FI^{3}において，各コードが担当する計算領域と生成されるプラズマ密度プロファイル．

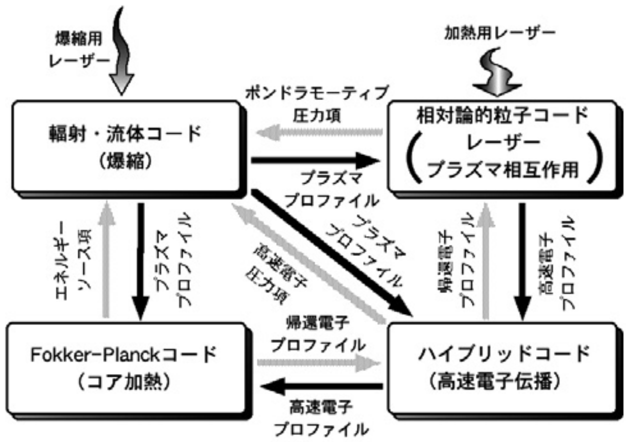

Fig. 2 FI^{3}におけるコードの結合ダイヤグラム．

レーザーとプラズマの相互作用は，現象を忠実に記述できる相対論的粒子コード[9]を用いる．さらに，高速電子の伝播は，プラズマの密度が固体密度に達するので粒子コードで直接計算することはコンピュータの能力上困難なため，バックグラウンドプラズマは流体として扱い，高速電子のみを粒子として扱うハイブリッドコード[10]を用いる．最後の高速電子のエネルギー緩和と原子核反応については，Fokker-Planck コード[11]により精密に計算する．以上述べた4つのコードがそれぞれ担当する計算領域の模式図を生成されるプラズマの密度プロファイルとともにFig. 1に示す．

FI^{3}は，超高強度レーザーと物質の相互作用を総合的に解析できるので，レーザー核融合分野に留まらず，レーザー核・素粒子物理やレーザー加速等の学術研究のほか，ペタワットレーザーで生成する高速粒子を用いたガンの治療・診断などの医療等の応用研究への活用も期待できる．

2.2 コード結合とデータ交換

FI^{3}において，各コードがシミュレーションする物理モデルの詳細は，3章で述べられるので，本節ではコード間の結合のみに焦点を当てる．各コード間でやり取りすべきデータをコード結合ダイヤグラムとしてFig. 2に示す．

高速点火における爆縮の時間スケールと加熱の時間スケールの違いを考慮すると，相対論的粒子コード，ハイブリッドコードおよびFokker-Planck コードのシミュレーション結果が，輻射・流体コードの実行状態に大きな影響を与えることはないと考えられるので，統合シミュレーションシステムの最初の段階では，Fig. 2の黒矢印で示す一方向のデータ交換のみを考えている．まず，輻射・流体コードは，爆縮用レーザーの吸収および輻射を含む爆縮過程をシミュレーションし，最大圧縮時におけるコロナ領域から圧縮されたコアまでのプラズマプロファイルを保存する．加熱用レーザーは，500ミクロン程度まで膨張したコロナ領域のプラズマと相互作用するが，このような大きなプラズマ全体を粒子コードで扱うのは困難である．また，コア周辺のプラズマは，臨界密度の10^5倍にまで圧縮されると予想されるが，このような高密度プラズマも粒子コードで扱うのは困難である．そこで，相対論的粒子コードは，輻射・流体コードによって保存されたプラズマプロファイルから，レーザープラズマ相互作用で重要だと考えられる臨界密度の1/10程度のコロナプラズマから臨界密度の100倍程度のコア周辺プラズマまでのプロファイルを読み込み，シミュレーションシステムに初期プラズマとして導入する．このようにして，現実的なプロファイルを持つプラズマと加熱用超高強度レーザーの相互作用をシミュレーションすることが可能になる．相対論的粒子コードは，レーザープラズマ相互作用をシミュレーションし，その結果生成された高速電子について，高密度側境界における詳細なプロファイルを決められた時間ごとに定期的に保存する．ハイブリッドコードは，輻射・流体コードによって保存されたプラズマプロファイルから粒子コードでは扱えない高密度プラズマのプロファイルを読み込み，それを背景プラズマとしてシミュレーションシステムに導入する．さらに，相対論的粒子コードによって書き出された高速電子プロファイルを読み込み，それを時間に依存した高速電子のソース項として扱い，高速電子の伝播をシミュレーションする．そして同様に，コアに達する高速電子についての詳細なプロファイルを決められた時間ごとに定期的に保存する．最後にFokker-Planck コードは，輻射・流体コードによって保存されたプラズマプロファイルからコアプラズマのプロファイルを読み込み，それを背景プラズマとするとともに，ハイブリッドコードから受け取った高速電子プロファイルを時間に依存した高速電子のソース項として扱い，高速電子によるコア加熱を計算する．以上述べたコード結合におけるデータ交換にはフィードバックがなく，このためデータ交換に伴う数値的な不安定性は発生しない．

統合シミュレーションシステムの次の段階では，双方向性のデータ交換を計画しており，より長時間スケールの現象を扱う予定である．即ち，相対論的粒子コードによってシミュレーションされた加熱用レーザーによるポンドラモーティブ力およびハイブリッドコードによってシミュレーションされた高速電子の動圧を圧力項として輻射・流体コードにフィードバックする．また，Fokker-Planck コードによってシミュレーションされた帰還電子のプロ

ファイルをハイブリッドコードおよび相対論的粒子コードの背景電子として反映する．さらに，コア中で起こる原子核反応をFokker-Planckコードで計算することによって，その効果をエネルギーのソース項として輻射・流体コードにフィードバックし，流体運動を補正する．なお，相対論的粒子コードとハイブリッドコードおよびハイブリッドコードとFokker-Planckコードは境界面で結合されているが，この統合段階の結合のようにフィードバックがある場合には，結合境界面において一般的に非物理的な現象が発生する．このため，この部分の結合は，単純な境界結合とはせずに計算領域に重複する部分を設けて容積結合とする新しい計算手法を開発する必要があると思われる．

2.3 データ交換のための専用プロトコル

FI^{3}のコード間におけるデータ交換は，コンピュータ上で実行されているプロセス間の通信になるため，基本的には並列計算においてプロセス間通信に利用されているMPIを用いて記述できる．しかし，MPIはSingle Program Multiple Data特性であることを考えると，Multiple Program Multiple Data特性であるFI^{3}のコード間通信をMPIでプログラミングすることは難しい．さらに，各コードそのものは並列化されていることが前提になるが，この並列化手法にMPIを用いている場合，コード間通信とコード内並列化の両方をMPIでプログラミングしなければならず，いっそうの困難さを伴う．また，各コードは独立して開発され，コードごとに適したアーキテクチャのコンピュータ上で実行することを想定しているので，データ交換はサイト間に及ぶ．しかし，このような環境下でMPI通信を許可することはセキュリティの低下を招くので，計算機システムの運用上非常に厳しい．このため，コード間通信をMPIで実装することは現実的ではない．一方，サイト間を含むコード間連携を行うためにグリッドの基盤ソフトであるGlobus，分散環境上のオブジェクト間でメッセージを交換するためのCORBA，XMLとHTTPをベースにしてネットワーク上にあるデータやサービスを呼び出すSOAP等を利用することは可能である．しかし，これらのインターフェイスやプロトコルは，汎用的な機能の提供を目的としており，ユーザに対して煩雑かつ複雑なプログラミングを要求する．その他に，交換すべきデータの通信を通常のファイル転送によって実装するが，スクリプティングによってユーザにはその動作を隠蔽し，あたかもコード間で直接データ交換しているように見せかけることも可能であり，ITBL基盤ソフトウェアで採用されている．また，米国ローレンスリバモア国立研究所では，オブジェクト指向スクリプト言語Pythonを用いて統合シミュレーションをこの方法で実装している．しかし，この手法では，FI^{3}の次段階で計画している双方向にデータ交換を行う場合，コード結合を実装するのは非常に煩雑になり，コードの大幅な改修も必要とする．

そこで，FI^{3}におけるコード間のデータ交換には，複雑な機能が必要ではないことを考慮し，軽装な専用プロトコルとして分散コンピューティング連携プロトコル（DCCP: Distributed Computing Collaboration Protocol）を設計・開発・実装し，高速ネットワークを介しての複数コードの統合動作を可能にする[12,13]．DCCPは，TCP/IPを用いた上位のプロトコルであり，通信専用の独自システムを構築し，サイト内では通常のLANが，サイト間ではインターネットが利用できるため，適用可能範囲は非常に広い．

FI^{3}では大規模シミュレーションを前提にしているので，個々のコードのジョブはサイトの計算機センターにある大型コンピュータで実行されることが多いと考えられる．このような大規模な計算資源は，多数のユーザが共有しているため，一般にジョブの実行はNQS（Network Queuing System）により管理されている．通常，TCP/IPを用いた通信を行うためには，送信元および受信先コードが同時に実行されており，かつ，受信先コードが実行されているコンピュータのIPアドレスが，送信元コードに対して既知でなければならない．しかし，NQSの管理下では，ジョブは一旦待ち行列に入れられるため，データを送信，受信するコードのジョブを同時に投入しても，両方のコードが同時に実行されるとは限らない．また，両コードが同時に実行されたとしても，NQSではアイドル状態となった資源に待ちジョブを動的に割り当てるため，送信元コードのジョブ投入時には，受信先コードが実行されるコンピュータを一意的に決定できない．したがって，ジョブの実行管理にNQSが使われている場合，統合シミュレーションに必要なデータ交換は，コード間の直接TCP/IP通信によっては実現できない．この問題を解決するために，DCCPでは，アービトレータ，コミュニケータと呼ぶ2種類のデーモンをネットワークに常駐させて通信を行い，コード間では直接通信を行わない構成をとる．アービトレータとはコード間の通信を管理するデーモンであり，DCCPを利用する全ネットワークに1つしか存在してはならない．また，コミュニケータとは実際にデータ通信を行うデーモンであり，DCCPを使用する各サイトに1つ以上存在するものとする．実際にコード間でデータ交換を行う場合，まず，送信元コードは送信側コミュニケータにデータを送信する．次に，送信側コミュニケータは，アービトレータに受信側コミュニケータを問い合わせ，送信元コードから受信したデータを受信側コミュニケータに転送する．そして，最終的に受信先コードが受信側コミュニケータにデータを要求・受信して，データ交換は完了する．このため，データ送信元コードは送信側コミュニケータのIPアドレスだけを知っていればよく，データ受信先コードのIPアドレスを知っておく必要はない．また，データ受信先コードがまだ実行されておらずデータを受信できない場合でも，送信元コードのデータ送信作業は完了し，そのデータはDCCPシステムのストレージに一時保存され，将来の転送に備えられる．そして，受信先コードが実行され，データが要求されると，DCCPシステムによりそのデータが自動的に読み出されて転送される．Fig. 3にDCCPの概念図を示す．Fig. 3において，制御通信とはコードとコミュニケータの管理およびコミュニケータ間のデータ転送を制御するために行われる通信であり，データ通信とはコード

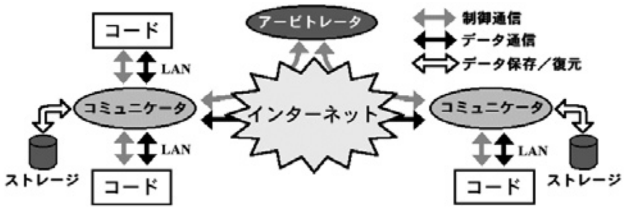

Fig. 3　DCCP の概念図.

間のデータ交換に必要なデータ転送である.

昨今のインターネットセキュリティの問題から，各サイトはファイヤウォールを導入していることが多く，ファイヤウォールによってDCCPの通信が遮断される可能性が高い．この場合には，トランスポンダと呼ぶ3番目のデーモンをファイヤウォール上に常駐させる．トランスポンダは，ファイヤウォール内のコミュニケータからは外部のアービトレータおよびコミュニケータに見え，外部のアービトレータ，コミュニケータからはファイヤウォール内に存在するコミュニケータに見える．このため，DCCP の制御通信およびデータ通信は，トランスポンダによって透過的に中継され，ユーザはファイヤウォールの存在に気をかけることなくデータ交換が可能になる．以上述べた機能を実現するための具体的なプロトコルを規定し，DCCP を実装した．その開発履歴を Fig. 4 に示す.

最後に，DCCP の Fortran サブルーチンを Fig. 5 に示す．交換すべきデータは，コード固有の名前であるコード名（CodeName），同一のコードで実行される異なったシミュレーションパラメータを区別するためのラン名（RunName）および実行中の時間ステップ等を区別するためのタグ（Tag）で指定する．このデータの3レベル指定により，コード間の柔軟なデータ交換が可能になる．ユーザは，この5種類のサブルーチンを呼び出すだけで統合シミュレーションシステムにおける前述した機能を持つデータ交換が可能であり，プログラミングは非常に容易である.

2.4　まとめ

マルチスケールの複雑現象を全体的に理解するためには，階層的に積み上げられた多数のコードから成る統合されたシミュレーションシステムの構築が必須である．この多階層統合シミュレーションシステムを実現するために，コード間のデータ交換に特化した軽装な専用プロトコル DCCP を提案，実装した．DCCP では，データ交換に必要な手続をできる限り簡略化しているので，ユーザは簡単な通信プログラミングを行うだけでコードにデータ交換機能を実装できる．この DCCP を用いた統合シミュレーションにより，高速点火の実験結果と直接比較しうる現実的な条件下でのシミュレーションが可能になると思われる.

2003年9月：テストバージョンのリリース
2004年3月：α バージョンのリリース
　・Endian 問題の解決
2004年6月：β バージョンのリリース
　・動作の安定化
　・同一データの再利用
2004年8月：バージョン1.0のリリース
　・中継機能の改良
　・エラー処理の改良
2005年1月：バージョン1.1のリリース
　・バグフィックス
　・通信の効率化
2005年3月：バージョン1.2のリリース
　・マイナーバグフィックス

Fig. 4　DCCP の開発履歴.

```
CALL DCCP_INITIALIZE ( CodeName, RunName, Cnd )
CALL DCCP_SEND ( DstCodeName, DstRunName, N, Type,
                                    Data, Tag, Cnd )
CALL DCCP_UPLOAD ( NDst, DstCodeNames, DstRunNames,
                  N, Type, Data, Tag, ExpireHour, Cnd )
CALL DCCP_RECEIVE ( SrcCodeName, SrcRunName, N, Type,
                          Data,Tag, TimeoutSec, Cnd )
CALL DCCP_FINALIZE ( Cnd )
```

Fig. 5　DCCP の Fortran サブルーチン.

参 考 文 献

[1] M. Tabak, J. Hammer, M.E. Glinsky, W.L. Kruer, S.C. Wilks, J. Woodworth, E.M. Campbell, M.D. Perry and R. J. Mason, Phys. Plasmas **1**, 1626 (1994).
[2] 三間圀興，長友英夫，坂上仁志：プラズマ・核融合学会誌 **79**, 489 (2003).
[3] H. Sakagami and K. Mima, Laser Part. Beams **22**, 41 (2004).
[4] HPF2.0 公式マニュアル（スプリンガーフェラーグ，東京，1999）.
[5] http://www.openmp.org/.
[6] http://www.mpi-forum.org/.
[7] 坂上仁志，長友英夫，城崎知至，三間圀興：レーザー研究 **32**, 324 (2004).
[8] H. Nagatomo, N. Ohnishi, K. Mima, K. Sawada, K. Nishihara and H. Takabe, *Proc. 2nd Int. Conf. Inertial Fusion Sciences and Applications*, Kyoto, 2001 (Elsevier, Paris, 2002) p.140.
[9] H. Sakagami and K. Mima, *Proc. 2nd Int. Conf. Inertial Fusion Sciences and Applications*, Kyoto, 2001 (Elsevier, Paris, 2002) p.380.
[10] T. Taguchi, T.M. Antonsen Jr. and K. Mima, Comput. Phys. Commun. **164**, 269 (2004).
[11] T. Johzaki, K. Mima, Y. Nakao, T. Yokota and H. Sumita, Fusion Sci. Technol. **43**, 428 (2003).
[12] 坂口智哉，坂上仁志，新居　学，高橋　豊：電子情報通信学会技術研究報告 **103**, 61 (2003).
[13] T. Sakaguchi, H. Sakagami, M. Nii and Y. Takahashi, *Parallel and Distributed Computing: Applications and Technologies*, LNCS 3320 (Springer-Verlag, Heidelberg, 2004) p.90.