対話的動作を考慮したタンパク質立体構造予測システム

蟻川 浩[†,☆] 増田 慎吾[†] 古田 忠臣[††,☆☆]
金 文珍[††] 朴 聖俊[††] 高田 彰二[††]
藤川 和利[†] 砂原 秀樹[†]

本論文では，対話的動作を考慮したタンパク質立体構造予測システムとその適用事例について述べる．タンパク質立体構造予測の現場では，エキスパートの経験を導入した試行錯誤によるプログラムの実行が求められている．そこで，我々は試行錯誤によるプログラムの実行を実現するために，ワークフローを修正すると修正したワークの位置から再実行するといった，柔軟なワークフロー制御を導入した対話的動作を考慮した計算システムを提案する．そして，提案したシステムに ROKKY と呼ばれるタンパク質立体構造予測システムを対話的動作を考慮した計算システムとして適用した．我々の提案するシステムを使うことにより，利用者は計算システムを意識することなくタンパク質立体構造予測を行えるようになった．

A Computing System for Protein Structure Prediction with Trial-and-error Process

Hiroshi Arikawa,[†,☆] Shingo Masuda,[†] Tadaomi Furuta,[††,☆☆]
Wenzhen Jin,[††] Sung-Joon Park,[††] Shoji Takada,[††]
Kazutoshi Fujikawa[†] and Hideki Sunahara[†]

The paper describes a computing system for a protein structure prediction and a case study of its system. Scientists of protein structure prediction have to do execution applications for protein structure prediction by trial-and-error process. For the purpose of doing execution applications by trial-and-error process, we propose a computing system with a flexible workflow mechanism, and apply a protein structure prediction system called ROKKY. By using the system, scientists could be easily to do simulation of protein structure prediction.

1. はじめに

計算機の高性能化およびネットワークの広帯域化にともない，インターネットに接続されている計算機を Globus Toolkit[1] などのツールキットによって仮想的な 1 つの高性能計算機として実現する手法としてグリッドコンピューティングがある．グリッドコンピューティングは当初，科学技術計算やデータ解析といった膨大な計算を必要とする分野で積極的に活用されてきた．グリッドコンピューティングにおける研究開発が進むにつれて，パラメータサーベイ，モンテカルロ法によるシミュレーションのように計算対象が持つ解空間を分割して計算したとき，その分割がそれぞれと疎な関係で，データの交換がほとんど行われないタスクを大量実行して解を得る方法，いわゆる高スループット計算が適していることが明らかになってきた．

我々はバイオサイエンスにおけるアプリケーションのグリッドコンピューティングへの適用，とりわけ，タンパク質立体構造予測のグリッドコンピューティングへの適用について，研究開発を行ってきた．

タンパク質立体構造予測は，ポストゲノム時代において遺伝情報の 1 つであるタンパク質の機能，構造を理論的に予測することである．タンパク質は我々の体を作るための主たる分子群であり，生命の組織や活動はタンパク質によって決められている．タンパク質の

† 奈良先端科学技術大学院大学情報科学研究科
Graduate School of Information Science, Nara Institute of Science and Technology
†† 神戸大学理学部化学科
Department of Chemistry, Faculty of Science, Kobe University
☆ 現在，関西大学経済・政治研究所政策グリッドコンピューティング実験センター
Presently with Policy Grid Computing Laboratory, Institute of Economic and Political Studies, Kansai University
☆☆ 現在，科学技術振興機構
Presently with Japan Science and Technology Agency

配列情報からコンピュータ・シミュレーションによって立体構造の予測が可能になれば，病気に対する新たな治療方法への検討，創薬への応用に発展することができる．このように，生命科学の応用面において，タンパク質立体構造予測手法の確立が大いに期待されている．

現在のグリッドコンピューティング技術において，グリッドコンピューティング向けのミドルウェアが多数存在する．ミドルウェアを使うことでタンパク質立体構造予測における大規模な演算処理を安価で実現できるようになった．一方で，ミドルウェアを使うためには，これらが提供しているコマンドを操作することが要求される．タンパク質立体構造予測の現場では，利用者の経験や知識を導入したシミュレーションの実施が望まれているが，コマンドによる操作は利用者が思い描いているプログラム実行の戦略を適用するのが困難である．また，入力パラメータを変更してプログラムを再実行するといった動作が頻繁に発生する．このように，利用者の試行錯誤の動作に基づいた処理を実現するためには，既存のミドルウェアだけでは不十分であることが研究開発を進めた結果，明らかになった．

我々はタンパク質立体構造予測のグリッドコンピューティングへの適用に際し，プログラムに与えるパラメータを利用者の知識によって反映できる計算システムを構築することを目標としている．本論文では，タンパク質立体構造予測用アプリケーションのプログラム実行に注目し，試行錯誤の動作に基づいたプログラム実行を支援する計算システムを提案する．

まず，利用者の対話的動作，特に試行錯誤の動作について PDCA サイクルと関係が深いことに触れたのち，PDCA サイクルの概念に基づいて構築したタンパク質立体構造予測システムについて述べる．そして，実際に構築したシステムを用いて，タンパク質立体構造予測を行い，試行錯誤の動作が予測結果に良い影響を及ぼすことを述べ，システムの有用性を示す．

本論文の構成は次のとおりである．2 章ではタンパク質立体構造予測の概要とタンパク質立体構造予測システム ROKKY について説明する．3 章では ROKKY を例としたタンパク質立体構造予測における試行錯誤の動作の必要性について示し，4 章で要求を満足するための予測システムの設計と実装について述べる．5 章では，構築した予測システムで実際にタンパク質立体構造予測を行ったときの結果を示す．6 章で関連研究について触れたのち，7 章で本研究のまとめおよび今後の課題について述べる．

2. タンパク質立体構造予測

2.1 予測手法の分類

タンパク質は約 20 種類のアミノ酸がペプチド結合によってつながった高分子であり，三次元立体構造に折りたたまって生化学的機能を発現する．タンパク質立体構造予測とは，X 線解析などにより立体構造が同定されていないアミノ酸配列を用いて，その配列がとりうる立体構造をモデリングすることであり，タンパク質立体構造構築原理や機能解析の理解にきわめて重要なアプローチである[2]．

立体構造未知のアミノ酸配列（以後，予測配列と呼ぶ）から立体構造を予測する方法は以下に分類される．

比較モデリング アミノ酸配列の配列類似度が 30%以上の 2 つのタンパク質は進化的に相同な近縁関係にあり，類似の立体構造をとる．予測配列と近縁で立体構造既知のタンパク質を検索し，その結果に対して立体構造テンプレートに予測配列をはめ込むことができる．このような方法を比較モデリング（**Comparative Modeling，CM**）と呼ぶ．

フォールド認識 立体構造テンプレートが CM の配列相同検索によって発見できない場合，フォールド認識（**Fold Recognition，FR**）法を用いて遠縁の構造既知タンパク質を検出し，CM を行う．

新規フォールド 近縁および遠縁の立体構造テンプレートが存在しない予測配列に対して，第一原理に基づき物理化学的エネルギー関数による立体構造を予測する方法（de novo 予測法）を新規フォールド（**New Fold，NF**）と呼ぶ．

1 つの予測配列が複数のドメインから構成されている場合がある．一般的に，このような予測配列はドメインに分割されたあと，それぞれのドメインに適した予測手法がとられる．

本論文では配列相同検索と FR によってテンプレートに基づく立体構造予測が可能な配列をそれぞれ CM ターゲット，FR ターゲットと呼び，それ以外を NF ターゲットと呼ぶことにする．

2.2 全自動立体構造予測システム ROKKY

ROKKY[3),4)] は，CM ターゲット，FR ターゲットおよび NF ターゲットを自動的に判別してタンパク質の立体構造を予測するシステムである．このシステムは国際的立体構造予測コンテスト CASP6（6th Critical Assessment of Techniques for Protein Structure Prediction）において FR および NF ターゲットの予測精度がサーバとして世界 2 位の実力を持つ高性能な全自動予測システムである．

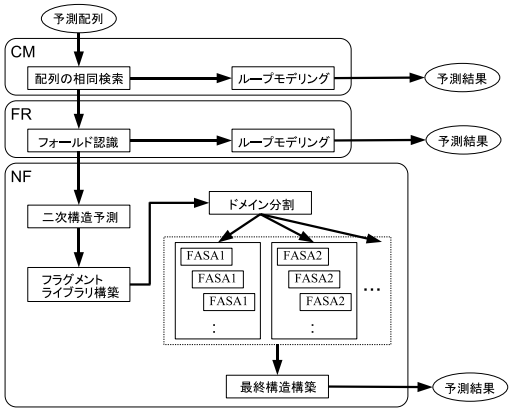

図 1 ROKKY における処理の概略
Fig. 1 Overview of the process for ROKKY.

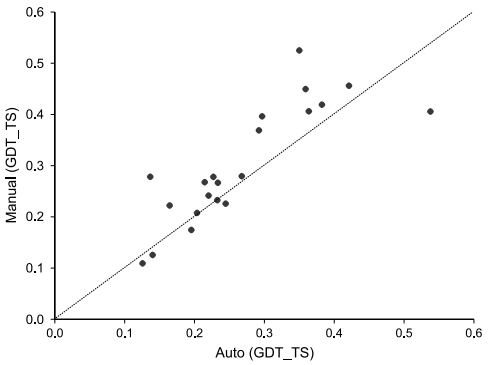

図 2 全自動予測結果と手動予測結果の関係
Fig. 2 The correlation between prediction result by full automation and by human intervation.

ROKKY は，PSI-BLAST [5] と FUGUE [6] によるテンプレート検索のツール群と de novo 予測の Fragment Assembly Simulated Annealing（FASA）を用いた SimFold [7),8)] から構成されている．ROKKY における処理の流れを **図 1** に示す．ROKKY の立体構造予測は PSI-BLAST による予測配列の相同検索を行うことから始まる．統計的有意な近縁のテンプレートが検出できなかった場合は FUGUE によるフォールド認識を行って遠縁のテンプレートを検索する．見つかった近縁・遠縁のテンプレートを用いて比較モデリングを行う．テンプレートが存在しない短い領域に対しては構造既知データベースを用いたループモデリングを行う．テンプレートが存在しない予測配列は NF ターゲットとして FASA を行う．まず，テンプレート検索結果を用いてドメイン分割を行う．そして，それぞれのドメインに対して FASA を複数回実行し，各試行の予測結果をクラスタ分析することによって最終的な予測構造を構築する．

3. タンパク質立体構造予測における試行錯誤の動作に関する要求要件

3.1 タンパク質立体構造予測における試行錯誤過程の重要性

現在，タンパク質立体構造予測手法を確立するための研究がさかんに行われている．研究者は予測手法の確立に向けて，より天然構造に似た結果を得るための方法を模索をしている．

図 2 は CASP6 において出題された予測配列のうち 21 個の予測配列について，ROKKY を使って得た予測結果（全自動による予測）と研究者が計算結果を見ながら手動で操作して得た予測結果（手動による予測）との関係を示したものである．後者はプログラムの実行結果を見ながら研究者らが持っている知識や経験をもとにプログラムに与えるパラメータを修正して得た結果である．横軸と縦軸はそれぞれ全自動予測結果と手動予測結果の GDT_TS（Global Distance Test Total Score）[9] である．GDT_TS は誤差 xÅ($x = 1.0, 2.0, 4.0, 8.0$) 以下で天然立体構造と重ねられる予測立体構造の残基の割合 GTD_P_x によって下式のように定義される．

$$GDT\_TS = (GTD\_P_1 + GTD\_P_2 + GTD\_P_4 + GTD\_P_8)/4$$

図 2 では 7 割の予測配列が破線よりも上側に分布している．この結果から，予測配列の多くは依然として手動による予測が有効であるとうかがえる．タンパク質立体構造予測においてより天然構造に近い結果を得るためには，試行錯誤の動作に基づいてパラメータの修正を行うことが重要である．

3.2 試行錯誤の動作を実現するための要求要件

タンパク質立体構造予測手法の研究者がより良いタンパク質立体構造予測結果を得るためにグリッドなどの計算環境を利用する際，試行錯誤によってプログラムに与えるパラメータを修正できることが求められる．具体的には以下の項目を満たすことが計算環境に求められる．

- **プログラムに与える初期入力パラメータの定義**
  シミュレーション全般において，何らかの計算を行う際，実行するプログラムに対して様々な初期入力パラメータを設定する．したがって，プログラムに与える初期入力パラメータの定義が容易にできることが必要である．
- **プログラムおよびプログラム実行順序の定義**
  通常，シミュレーションでは前処理，本処理，後

処理といった形で複数のプログラムを用いる．実行するプログラムおよびプログラムの実行順序を決め，その順序に従ってプログラムを実行する仕組みが必要である．

- **計算中に結果を随時見ながらのパラメータ修正**
  たとえば，ROKKY で用いられている FASA のようなパラメータサーベイの計算手法では，得られる結果が容易に想像できないため，解の導出のためには考えられる範囲ですべての計算を行う．解を導出するための計算は非常に長い時間を必要とするため，利用者が持っている知識や経験を試行錯誤の動作として取り入れて，目標とされる結果へと誘導することが必要になる．そのためには，計算中に結果を随時見ながらパラメータを修正できることが必要である．

4. 対話的動作を考慮したタンパク質立体構造予測システム

我々は利用者の経験によるパラメータなどの変更を可能にするためには，利用者と計算システムとの間に対話的な動作をする機能が必要だとの認識から，既存のミドルウェアで構築した計算環境に対して，PDCA サイクルの動作をふまえたタンパク質立体構造予測を支援するための計算システムを提案する．

4.1 試行錯誤動作のモデル

ROKKY を用いたタンパク質立体構造予測では，CM および FR によるデータベース検索エンジンの選択，NF においては，適切なパラメータを定義していかに多くの処理を行うかが重要である．この行動はタンパク質立体構造予測を行ってきた利用者の経験と理論的な知識，すなわちエキスパートの知識から決定される．そこで，利用者の試行錯誤による動作をモデル化したものとして PDCA サイクルがあげられる．

PDCA サイクルとは，ISO9001[10] における品質および顧客満足を向上させるための手法の 1 つで，効果的な品質管理を行うための手順を示したものである．具体的には，**図 3** に示すように，ある目標に対して，目標を達成できるような「計画（Plan）」を立て，「実行（Do）」し，実行結果を「確認（Check）」したのち，計画が正しかったかどうかを「見直し（Action）」する．そして，これらを繰り返しながら目標へと近づけていく．

タンパク質立体構造予測では，目標は正しいとされる予測配列の発見であり，これに向かって，

Plan 実行するプログラムを選定し，プログラムの実行順序を決定する．また，プログラムに入力するパラメータを定義する，

Do 実行順序に基づいてプログラムを実行する，

Check 実行されたプログラムから得られた結果を回収し，結果を分析する，

Action 分析結果から，プログラムの再実行を行うか，パラメータの修正が必要かどうかを判断する，

を繰り返し行う．本論文では，PDCA サイクルをモデルに対話的操作を考慮にいれたタンパク質立体構造予測システムを提案する．

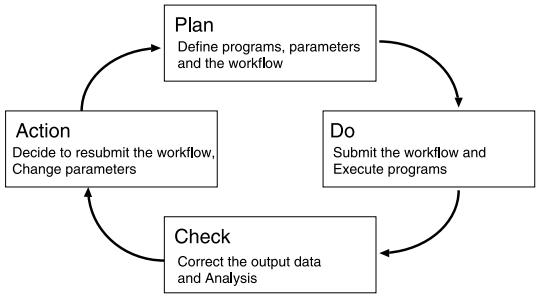

図 3 PDCA サイクル
Fig. 3 PDCA cycle.

4.2 設計方針

4.2.1 ワークフローの適用

我々が提案するタンパク質立体構造予測システムでは，利用者が持っている知識を試行錯誤の動作として取り入れるために，ワークフローに基づいたシミュレーション支援パッケージを提案する．

本論文では，利用者によって選択された実行プログラム名および各プログラムが実行する際に必要なパラメータで構成されるものをワーク，そして，ワークの実行順序をワークフローと定義する．便宜上，ワークとその実行順序を示したものをワークフローデータと呼ぶ．

ワークフローの概念を適用することで，シミュレーションにおけるプログラムの実行順序と実行するプログラムの内容が明確になる．このことから，ワークフローの概念は利用者が計算システムを利用する際の試行錯誤の動作を支援することが可能になる．

4.2.2 支援パッケージの構成

支援パッケージはワークフローによる利用者のプログラム実行戦略を実現できるように，既存のグリッドミドルウェアと以下に示す 4 つのモジュールで構成される．グリッドミドルウェアとモジュールの関係を**図 4** に示す．

プログラム実行モジュール タンパク質立体構造予測で用いるプログラムの実行を行うモジュール．ワークフローデータに基づいてワークを実行する．

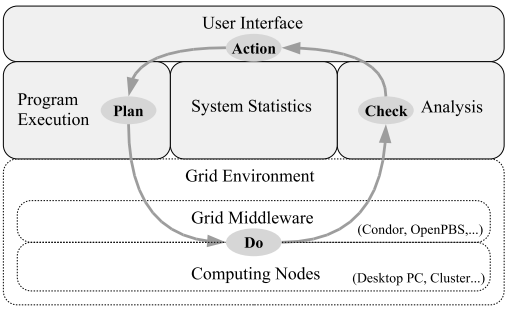

図 4 モジュール構成
Fig. 4 Overview of protein structure prediction computing system.

計算結果解析モジュール 各手法において出力された結果を回収し，計算結果の可視化を準備するためのモジュール．

システム状況監視モジュール ワークフローデータに基づいて実行されたワークの状況および計算システムを構成する計算ノードの負荷情報を収集するモジュール．グリッドミドルウェアに装備されているプログラムの実行状況確認コマンドおよび計算ノードの CPU 負荷情報確認コマンドを利用して情報を収集する．収集した情報はシステム状況監視モジュールが保持している．そして，Web ブラウザを利用して利用者へワークフローの進捗情報として提示したり，後述するワークフロー差分実行機構で差分ワークフローを作成する際の情報として用いられる．

ユーザインタフェース 利用者と計算システムとの架け橋の役目をするモジュール．ワークフローデータ記述の支援や計算結果の表示を行う．

我々が提案する計算システムでは，利用者の試行錯誤の動作をシステムの一部としてとらえている．したがって，提案するシステムではアプリケーション実行モジュールが Plan を，既存のミドルウェアで構築した計算環境が Do を，計算結果解析および表示モジュールが Check を，そしてユーザインタフェースを含む利用者の行動が Action を担当する．

4.3 ワークフロー差分実行機構の導入

試行錯誤の動作に基づいたシミュレーションを行う際，プログラム実行時に定義されるパラメータは計算結果を見ながら適宜変更することが想定される．従来，ワークフローデータに基づいたプログラム実行でパラメータの変更を行う場合，すでに動作中のワークフローデータを破棄した後，修正したワークフローデータによって再実行する手順であった．修正したワークフローデータによる実行は過去のワーク実行状況のいかんにかかわらずワークフローデータの先頭から開始されていた．

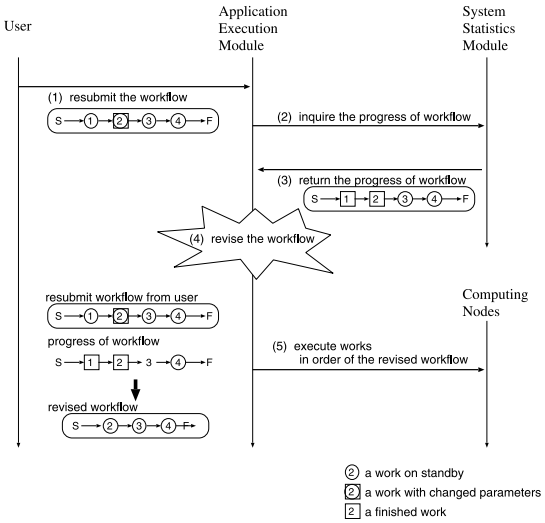

図 5 ワークフロー差分実行機構の処理の流れ
Fig. 5 Process of the workflow revise mechanism.

本論文では，PDCA サイクルにおける Action と Plan の動作に注目し，利用者による必要最低限の操作でワークフローの停止，破棄，再実行の操作を行う仕組みであるワークフロー差分実行機構を提案する．

ワークフロー差分実行機構は，ワークフローデータ，その中でもワークの情報修正に注目し，ワークの情報修正と連動して差分ワークフローを作成し，ワークフローの実行を継続する仕組みである．この機構により利用者はワークの情報修正とワークフローの再投入を行うだけでよい．

4.3.1 動作概要

図 5 はワークフロー差分実行機構の処理の流れを示したものである．処理の流れを説明する．

(1) 利用者が修正したワークフローデータ（修正ワークフローデータ）をプログラム実行モジュールに送信する．
(2) プログラム実行モジュールが受信した時点で，すでに実行されているワークフローの進捗状況をシステム状況監視モジュールに問い合わせる．
(3) システム状況監視モジュールはプログラム実行モジュールからの問合せに対して，問合せ時点でのワークフローの進捗状況（進捗ワークフローデータ）を回答する．
(4) システム状況監視モジュールから受け取った進捗ワークフローデータと修正ワークフローデータとを対照させ，新たに実行すべきワークフローの差分（差分ワークフローデータ）を作成

する．

(5) すでに実行されているワークフローを廃棄し，差分ワークフローデータに従ってワークの実行を行う．

4.3.2 ワークフローデータベース

利用者が作成するワークフローデータとワークフロー差分実行機構が保持するワークフローデータとの一貫性を保つため，ワークフローデータベースを準備する．ワークフローデータはプログラム実行モジュールがユニークに割り当てた識別番号（以後，ワークフロー ID と呼ぶ）で管理されており，ワーク情報およびワークフロー情報の 2 種類の情報をワークフローデータベースに登録する．

ワークに関する情報は

- ワークフロー ID
- ワーク番号
- 実行するプログラム
- プログラムに入力するパラメータ

で構成される．また，ワークフローは

- ワークフロー ID
- ワーク番号の列（ワークを実行する順序）

で構成される．

利用者は後述するワークフロー記述支援ツールを通じてワークフロー管理サーバにアクセスする．ワークフロー管理サーバはワークフローデータベースに問い合わせ，その結果をシステム利用者に送信する．ワークフローデータをもとに，プログラムに入力するパラメータを変更する．追加/削除によって更新されたワークフローデータから差分ワークフローデータを作成し，そのデータに従って計算処理を継続する．

4.3.3 差分ワークフローの作成

差分ワークフローの作成について説明する．

(1) 修正ワークフローデータとワークフローデータベースに登録されているワークフローデータとの比較を行い，ワークフローの修正箇所を検知する．そして，ワークフローの先頭から数えて最初となるワークフローの修正箇所（修正ワーク位置）を検知する．

(2) システム状況監視モジュールから進捗ワークフローデータを取得し，修正ワークフローデータが投入された時点で実行中のワークの位置（実行中ワーク位置）を検知する．

(3) 修正ワーク位置と実行中ワーク位置を比較し，差分ワークフローデータを作成する．

(a) 実行中ワーク位置が修正ワーク位置よりも前の場合，ワークフローデータベースに登録されているワークフローデータから実行中ワーク位置からのワークフローデータを作成したのち，修正ワークフローデータから得られる修正情報を使ってワークフローデータを作成する．

(b) 実行中ワーク位置が修正ワーク位置よりも後の場合，修正ワークフローデータから修正ワーク位置以後のワークで構成するワークフローデータを作成する．

4.4 ワークフロー記述支援ツールの導入

ワークフローの定義やワークフローに基づいたプログラムの実行といった操作を容易にするために，ワークフロー記述支援ツールを導入する．

ワークフロー記述支援ツールは GUI によるもので

- プログラムの実行順序および各プログラムのパラメータの定義
- ワークフローの投入，実行，停止処理

をマウス操作によって実現する．また，

- ワーク進捗状況の把握
- 任意のワークからの再実行

が可能である．

利用者がワークフロー記述支援ツールを使って入力する情報は，

- 実行するプログラムの名前
- プログラムを実行するために必要なパラメータ
- プログラムの実行順序

である．利用者はワークフロー記述支援ツールを使ってワークフローデータの作成を行う．そして，計算結果を確認しながらワークフローデータを修正し，ワークフローの再実行を行う．

4.5 システムアーキテクチャ

本論文で提案する計算システムは PC クラスタを含む計算ノード，ワークフロー管理部，プログラム実行制御部，計算結果解析部，ユーザインタフェース部で構成される．システム構成図を **図 6** に示す．

ワークフロー管理部 ワークフローの管理を行う．ワークフロー管理サーバには，ワークフロー差分実行機構が実装されている．ワークフロー管理サーバでは PostgreSQL を用いてワークフローデータベースを管理する．

プログラム実行制御部 ワークフローデータに基づいたワークの実行およびワークが適切に実行されていることを確認する．プログラム実行モジュールおよびシステム状況監視モジュールがメタジョブディスパッチャに実装されている．計算ノードへのジョブ実行は既存のミドルウェアを使用してい

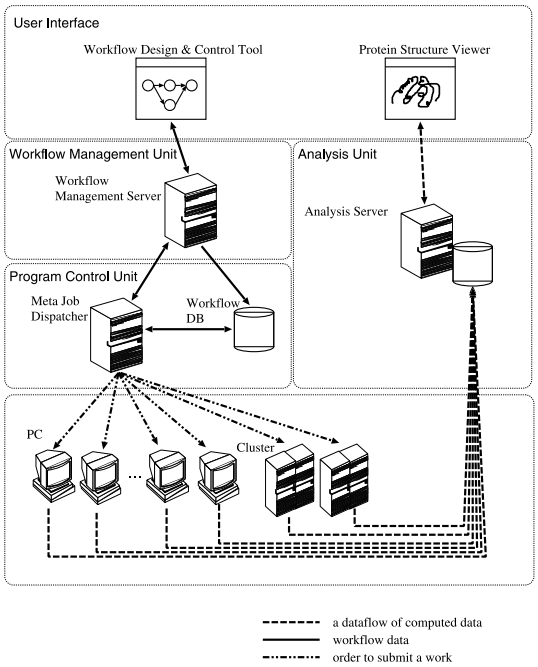

図 6 システム構成
Fig. 6 The computing system for protein structure prediction.

る．本論文では OpenPBS を採用した．

計算結果解析部 各計算ノードで計算した結果の回収および計算結果の可視化準備を行う．結果解析サーバには計算結果解析モジュールが実装されている．

ユーザインタフェース部 利用者が計算システムを操作するために必要なユーザインタフェースを提供する．利用者は **図 7** に示すワークフロー記述支援ツールを利用して，4.4 節で示した情報を入力する．また，計算結果は Web ブラウザを使って確認する．

提案するシステムを用いたときの処理の流れを以下に示す．

利用者はワークフロー記述支援ツールを利用して，ワークフロー作成に必要な情報を入力する．ワークフロー記述支援ツールから実行開始の操作を行うと，ワークフローデータが作成され，ワークフロー管理サーバに転送される．

ワークフローに基づいたプログラムの実行は，メタジョブディスパッチャがワークフローデータベースからワークフローデータを読み込み，各ワークの内容に基づいてワークの実行に適したノードを選択し，ワークが実行される．

プログラム実行結果は計算結果解析サーバが計算

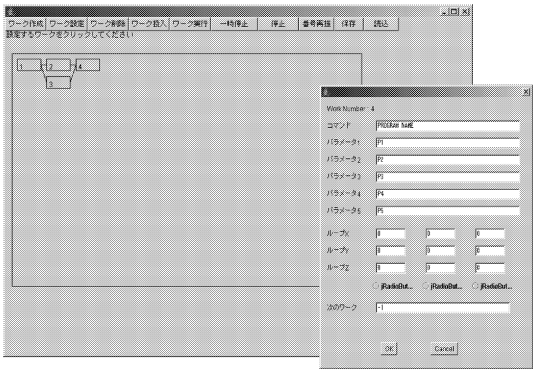

図 7 ワークフロー記述支援ツール
Fig. 7 Workflow design tools.

ノードから収集し，Web ブラウザで可視化できる状況に結果を加工する．

利用者は，適当な時間間隔で計算結果を見ながら，プログラムの実行順序や各プログラムの入力パラメータを検討する．出力された結果が利用者の希望と異なるものであれば，ワークフロー記述支援ツールを利用して各ワークの入力パラメータを修正し，ワークフロー管理サーバに通知する．

修正後のワークフローデータがワークフロー管理サーバに届くと同時に，ワークフロー差分実行機構が働き，ワークフロー管理サーバにおいて差分ワークフローデータを生成する．そして，メタジョブディスパッチャが差分ワークフローデータに基づいてワークを実行する．

5. 実行性能評価

5.1 計算機環境

本論文で提案するタンパク質立体構造予測システムを構築するにあたり使用した計算機の性能を 図 6 に対応させて説明する．

ワークフロー管理サーバおよびメタジョブディスパッチャのスペックはそれぞれ Intel 社 1.0 GHz Pentium III プロセッサ 1 基を搭載した PC，Intel 社 2.53 GHz Pentium 4 プロセッサ 1 基を搭載した PC を用いた．計算ノードは，独立した PC1 台と PC クラスタ 1 台を用いた．計算機のスペックは，それぞれ Intel 社 2.53 GHz Pentium 4 プロセッサ 1 基を搭載した PC，Intel 社 2.53 GHz Pentium 4 プロセッサ 1 基を搭載した PC を 1 台と Intel 社 1.0 GHz Pentium III プロセッサ 1 基を搭載した PC を 6 台で構成されたヘテロジニアスクラスタである．PC クラスタのジョブディスパッチャは Pentium 4 プロセッサ搭載の PC が担当する．計算結果解析サーバはワークフロー管理サーバとの兼

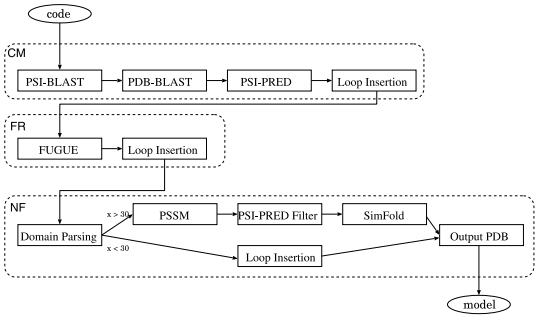

図 8 ROKKY におけるプログラム実行の流れ
Fig. 8 Workflow of ROKKY.

用とした．

ROKKY では 1 台でも多くの計算ノードを必要とするため，ジョブディスパッチャとなる Pentium 4 プロセッサ搭載の PC も計算ノードとしても利用できるように設定をした．すべての計算ノードは 100 Mbps の Ethernet で接続されている．

利用者からはワークが単独で実行できる 1 台の計算ノードと PC クラスタ 1 台，合計 2 台の計算ノードが登録されているように見える．

5.2 ワークおよびワークフローの準備

ROKKY は **図 8** に示すワークフローによってタンパク質立体構造予測が行われる．ROKKY では，CM，FR，NF の各手法を実行する Perl スクリプトと各手法を実行する順序が書かれた Perl スクリプトが存在する．前者はワークとして定義されたスクリプトであり，後者はワークフローが定義されたスクリプトと読み替えることができる．本論文では，CM，FR，NF の各手法を実行する Perl スクリプトはそのまま利用する．利用者はワークフロー記述支援ツールで CM，FR，NF 各ワークのパラメータとその実行順序を定義する．

各ワークを実行するための計算機の割当ては各ワークによって異なるため，以下のように計算機の割当てを定義した．CM および FR についてはデータベースを用いた検索を行うため，CM および FR で用いるデータを設置した 1 台の計算ノードに作業を割り当てる．NF については SimFold と呼ばれるプログラムが異なるパラメータを定義して大量に実行するため，PC クラスタに作業を割り当てる．

5.3 実行結果

本論文で提案する支援パッケージを導入した計算システムを用いてタンパク質立体構造予測を行った．

5.3.1 予測配列

入力となる予測配列は CASP6 において出題された問題の中から，Target T0198 および Target T0215 の 2 種類を用いた．T0198 および T0215 の特徴を以下に示す．

T0198 235 残基の予測配列問題である．ROKKY を用いた予測計算において，2 つのドメインに分割し，それぞれのドメインを SimFold を用いて立体構造予測を行ってしまうことが過去の実験で明らかになっている．2 つのドメインで計算しても希望どおりの予測結果が得られないので，希望どおりの予測結果を得るためには，1 つのドメインで SimFold を実行するようにワークの情報を修正する作業が必要になる．

T0215 76 残基の予測配列問題である．ROKKY を用いた予測計算において，SimFold の実行時に用いられるパラメータの 1 つである，フラグメントライブラリと呼ばれるデータが初期状態の場合では良い結果が得られないことが過去の実験で明らかになっている．希望どおりの予測結果を得るためには，修正したフラグメントライブラリで実行するようにワークの情報を修正する作業が必要になる．

このように，T0198 および T0215 は利用者の意図的な操作が必要な計算例である．本論文では T0198 および T0215 の天然構造は未知であると仮定して立体構造を求める実験を行った．

5.3.2 ワークフロー修正の動作シナリオ

利用者によるワークフロー修正の操作を行った場合のシナリオを以下に示す．

(1) CM，FR，NF の順に実行するワークフローデータを作成する．
(2) ワークフローデータに基づいたシミュレーションを行う．
(3) 適当な時間間隔で出力結果を確認する．
(4) 出力結果がふさわしくないと判断した時点でワークフロー記述支援ツールを用いてワークの情報修正をする．
- T0198 については，1 つのドメインで修正するように NF ワークの情報を修正する．
- T0215 については，すでに修正されたフラグメントライブラリを用いて実行するように NF ワークの情報を修正する．

(5) 修正したワークフローで再実行する．

一方，利用者による操作を行わなかった場合の動作のシナリオは (1)，(2)，(3) の操作だけである．

5.3.3 各ワーク単体の実行時間

T0198 および T0215 を予測配列として入力し，シ

表 1 CM および FR の平均実行時間（T0198, T0215）
Table 1 Result of the simulation time using CM and FR (T0198 and T0215).

	T0198	T0215
CM	3'26"	1'14"
FR	9'05"	5'32"

表 2 SimFold プログラム単体の平均実行時間（T0198）
Table 2 Result of the simulation time using SimFold (T0198).

残基数	111	124	235
Pentium III 1 GHz	2h32'00"	2h58'00"	9h44'00"
Pentium 4 2.53 GHz	1h20'00"	-	4h55'00"

表 3 SimFold プログラム単体の平均実行時間（T0215）
Table 3 Result of the simulation time using SimFold (T0215).

残基数	76
Pentium III 1 GHz	1h10'00"
Pentium 4 2.53 GHz	0h35'00"

ミュレーションを行ったときの CM と FR の平均実行時間を**表 1** に示す．また，T0198 および T0215 における SimFold 単体の平均実行時間をそれぞれ **表 2**, **表 3** に示す．表 2 については利用者による操作を行わない状態でシミュレーションを行うと，2 つのドメインに分割して SimFold を実行するため，残基数が 111 残基と 124 残基の 2 種類の結果もあわせて示した．

これらの結果から分かるように，残基数が増えると CM，FR，NF 各ワークは長い実行時間を要する．

5.3.4 ワークフロー修正によるシミュレーション結果の違い

提案する支援パッケージを導入したことによって，タンパク質立体構造予測のシミュレーション結果がどのように変化するかについて評価する．

図 9 および **図 10** は T0198 および T0215 のそれぞれにおいて，利用者によるワークフローデータの修正を行った場合と行わなかった場合でタンパク質立体構造予測を行ったときの正解との類似度の時間的変化を示したものである．横軸は時間を縦軸は最小二乗誤差（Root Mean Squared Distance，RMSD）を示す．RMSD は天然立体構造（正解）の i 番目原子と予測立体構造の j 番目原子との距離 d_{ij} を用いて下式のように定義される．

$$\mathrm{RMSD}=\sqrt{\frac{1}{n}\sum_{i,j=1}^{n} d_{ij}^2}$$

このとき，n は原子数を表す．

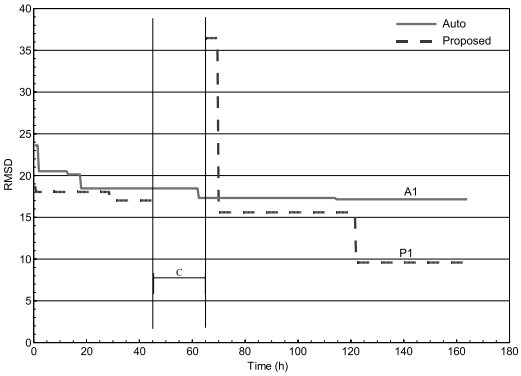

図 9 RMSD 最小値の時間的変化（T0198）
Fig. 9 Time-series change of minimum RMSD (T0198).

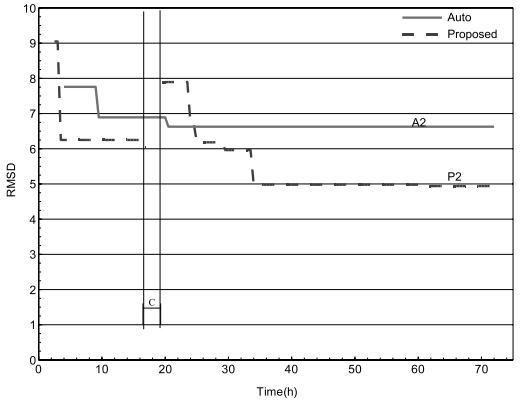

図 10 RMSD 最小値の時間的変化（T0215）
Fig. 10 Time-series change of minimum RMSD (T0215).

T0198 および T0215 のシミュレーションにおける RMSD の値が最も小さい立体構造をそれぞれ **図 11** および **図 12** に示す．図中 Auto はワークフローの修正を行わなかった場合，図中 Proposed はワークフローの修正を行った，つまり利用者の操作に基づいた場合の結果である．

T0198 では 2 つのドメインから 1 つのドメインで実行するようにワークフローの修正を行ったため，図 11 Proposed のように 1 つのドメインになり，正解の構造に近づいている．切り替えた時点では RMSD の値が 35.98 と良くないが，計算を続けていくうちに切り替えてから 55 時間後 10 以下になっている．一方，ワークフローの修正を行わなかった場合，2 つのドメインのまま計算を継続しているため，図 11 Auto のように α と β の部分に分離している．計算を継続しても RMSD の値が 17 を推移したままであることが 図 9 Auto のグラフから読み取れる．よって，これ以上計算を継続しても良い結果が得られないことが分かる．

また，T0215 では修正したフラグメントライブラリ

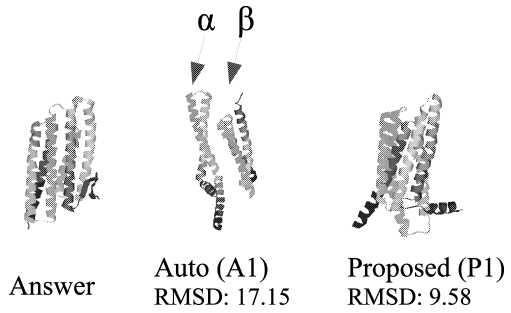

図 11 正解との立体構造の比較（T0198）
Fig. 11 Compare answer and computed data of protein strucuture prediction (T0198).

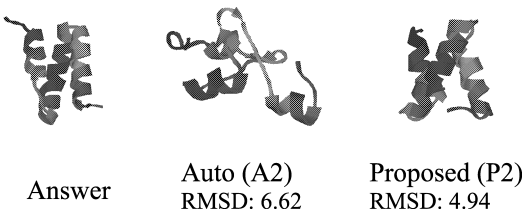

図 12 正解との立体構造の比較（T0215）
Fig. 12 Compare answer and computed data of protein strucuture prediction (T0215).

を用いるようにワークフローの修正を行ったため，図 12 Proposed の構造が正解の構造に似た構造になった．ワークフローデータの修正を行った時点では RMSD の値が 7.89 と良くないが，切り替えてから 14.5 時間後の 34 時間後には 5 以下になっている．一方，ワークフローの修正を行わなかった場合はフラグメントライブラリが初期状態のままなので，RMSD の値が 6.6 を推移しており，それ以下になる兆しが見られない．よって，図 10 から正解に似た構造が得られるとは考えにくい．

これらの結果から，提案するシステムを用いてワークフローデータの修正を行うと，立体構造の予測精度を改善できることが示せた．

5.3.5 ワークフロー修正によるシミュレーション実行時間の影響

ワークフロー差分実行機構の導入によって，利用者によるワークフローの修正が容易になったが，それにともないシミュレーション実行時間に影響を及ぼす．

利用者による操作を行った際，T0198 においては 45 時間から約 19 時間分の結果が，T0215 においては 17 時間から約 2 時間分の結果が出力されていないことが図 9 および 図 10 から分かる．それぞれの図中 C が結果が出力されていない時間帯に該当する．

これは，ROKKY の結果出力モジュールの処理上の制限によるものであることがすでに分かっている．ROKKY の結果出力プログラムにおいて，出力結果の統計処理（具体的には，クラスタ分析）を行う．統計処理を行うためには SimFold の結果が最低でも 10 個必要である．今回の実験では 7 台の PC で構成されたヘテロジニアスクラスタで SimFold を実行しているので，10 個分の SimFold の実行結果を得るためには各計算ノードで 2 回以上 SimFold の実行が必要になる．T0215 では修正前も修正後も同じ残基数のシミュレーションを行っているため，表 3 で示した Pentium III プロセッサ搭載の計算ノードでの SimFold プログラム実行時間の 2 倍とほぼ一致する．また，T0198 では 2 つのドメインから 1 つのドメインで実行するように切り替えたため，SimFold で計算すべきドメインの残基数が大きくなる．表 2 で示した 235 残基の結果から Pentium III プロセッサ搭載の計算ノードでの SimFold プログラム実行時間の 2 倍とほぼ一致している．

なお，ワークフロー差分実行機構そのもののオーバヘッドについて，修正したワークフローが反映されるまでの時間をワークフロー管理サーバで測定したところ 20 秒であった．

これらの結果から，ワークフローの修正によるシミュレーション実行時間の影響は予測配列の残基数に影響を受けることが分かった．

6. 関 連 研 究

我々の狙いは，予測手法の確立が実現するまでは，試行錯誤の動作に基づいたタンパク質立体構造予測を行い，そこで得られた知見をタンパク質立体構造予測の自動化につなげていくことである．そのためには，PDCA サイクルがスムーズに回転することが望ましい．

グリッドコンピューティング環境を構築するためのミドルウェアとしては，たとえば，Globus，Condor [11]，UNICORE [12] がある．これらのミドルウェアは他の計算機でプログラムを実行するための機能が提供されており，コマンドラインで操作コマンドを入力することが要求される．コマンドラインによる操作はパラメータやプログラム実行順序の変更といった利用者による試行錯誤の動作において，パラメータの誤入力が発生するといった問題が発生するため，ミドルウェアのみでシステム構築することは操作の観点で不十分である．なお，UNICORE では，利用者による操作を容易にするために GUI を用いたワークフローシステムが提供されているが，本論文で提案するワークフロー

差分実行機構のように，ワークフローデータの修正と連動した仕組みは用意していない．

利用者と計算機システムとの対話的な操作を実現する技術としてグリッドポータル（たとえば GridPort[13]）がある．グリッドポータルでは，グリッドコンピュティング環境を構築するためのミドルウェアが持つ複雑な操作を隠蔽するために，一般的には Web ブラウザを使って利用者との対話的な操作を実現している．あらかじめ登録されたプログラムに対してパラメータを入力し，実行する方式が採用されているため，利用者との対話的な操作を容易にしている点では優れているが，シミュレーションで使うプログラムを変更すると同時にポータルシステムの再構築が必要といった問題を含んでいる．我々はパラメータの変更以外にも，プログラムとそれを実行する順序が定義できることが望ましいと考えている．我々の要求はパラメータの変更だけでなくプログラムおよびその実行順序が変更できることである．そして，その行動によって出力結果が反映されることである．したがって，グリッドポータルでは我々の要求を満たさない．

計算機の遊休状態を利用したタンパク質立体構造予測の取り組みとして，CHARMM と呼ばれる分子動力学法を用いたプログラムのグリッドへの適用に関する研究[14]，および Predictor@home[15] があげられる．前者では，United Device 社の Meta Processor platform[16] で構築された計算機環境，後者では Berkeley Open Infrastructure for Network Computing（BOINC）[17] によって構築された計算機環境を利用して計算システムが構築されている．いずれの場合も，インターネットに接続可能な計算機の遊休状態を利用して，高速な演算処理能力を実現している．一方で，利用者の経験によるパラメータ変更の必要性については考慮していない．本論文で提案した予測システムの計算ノード群をこれらの計算機環境に対して適用することは高い演算処理能力を得られることなので，これらの計算機環境を考慮に入れたメタジョブディスパッチャの設計は興味深い課題である．

7. おわりに

タンパク質立体構造予測のグリッドコンピューティングへの適用に向けて，利用者が持つ過去の経験や知識を立体構造予測に反映することが可能なタンパク質立体構造予測システムを提案した．提案したシステムは利用者の試行錯誤の動作を念頭に置いた設計と実装であることを示した．そして，提案システムを用いて自動化システムでは解くことが難しいとされる予測配列に対して実際に立体構造予測を行った．実験に用いた計算機の処理能力が最近の計算機のそれに比べて低いこと，また，利用した計算機の台数が 7 台と少ないことから，利用者による操作に切り替えてから結果を得るために時間を要したものの，ワーク情報を修正することで，タンパク質立体構造予測の結果が良くなることを示した．試行錯誤の動作に基づいたタンパク質立体構造予測を実現したことで，対話的動作を考慮したタンパク質立体構造予測システムの有用性が確認できた．

我々の目標はグリッドコンピューティングにおけるタンパク質立体構造予測システムに関して，演算処理能力の向上を図りつつ，利用者にとって使いやすい環境を構築することである．本論文では 1 つのワークフローに対して利用者のプログラム実行戦略を反映できるようにした．今後は複数のワークフローに対して計算機資源の配分を視野にいれた利用者のプログラム実行戦略を反映できるシステムの構築について取り組んでいく．

参考文献

1) Foster, I. and Kesselman, C.: Globus: A Metacomputing Infrastracture Toolkit, *International Journal of Supercomputer Applications*, Vol.11, No.2, pp.115–128 (1997).
2) Baker, D. and Sali, A.: Protein Structure Prediction and Structural Genomics, *Science* (2001).
3) ROKKY Protein Structure Suite. http://www.proteinsilico.org/rokky/
4) Jin, W., Furuta, T., Park, S.J., Koga, N., Fujitsuka, Y., Chikenji, G. and Takada, S.: ROKKY: structure prediction server that integrates PDB-BLAST, 3D-Jury and the SimFold fragment assembly simulator, *CASP6 Abstracts*, pp.128–129 (2004).
5) Altschul, S.F., Stephen, F., Thomas, L.M., Alejadro, A.S., Jinghui, Z., Zheng, Z., Miller, W. and Lipman, D.J.: Gapped BLAST and PSI-BLAST: a new generation of protein database search programs, *Nucleic Acids Research*, Vol.25, pp.3389–3402 (1997).
6) Shi, J., Blundell, T.L. and Mizuguchi, K.: FUGUE: sequence-structure homology recognition using enviroment-specific substitution tables and structure-dependent gap penalties, *Journal of Molecular Biology*, pp.248–257 (2001).
7) Takada, S.: Protein Folding Simulation With Solvent-Induced Force Field: Folding Pathway

Ensenble of Three-Helix-Bundle Proteins, *Proteins: Structure, Function and Genetics*, No.42, pp.85–98 (2001).
8) Fujitsuka, Y., Takada, S., Luthey-Schulten, Z.A. and Wolynes, P.G.: Optimizing Physical Energy Functions for Protein Folding, *Proteins: Structure, Function and Bioinformatics*, Vol.54, pp.88–103 (2004).
9) Zemla, A.: LGA — A Method for Finding 3D Similarities in Protein Structures, *Nucleic Acids Research*, Vol.31, No.13, pp.3370–3374 (2003).
10) Quality management systems — Requirements, ISO9001. (2000).
11) Condor Project. http://www.cs.wisc.edu/condor/
12) UNICORE Forum. http://www.unicore.org
13) GridPort. http://gridport.net/index.cgi
14) Uk, B., Taufer, M., Stricker, T., Settanni, G. and Cavalli, A.: Implementation and Characterization of Protein Folding on a Desktop Computational Grid Is CHARMM a suitable candidate for the United Device MetaProcessor?, *Proc. International Parallel and Distributed Processing Symposium*, IEEE (2003).
15) Predictor@home. http://predictor.scripps.edu
16) United Devices, Inc: Edge Distributed Computing with the MetaProcessor Platform (2001). http://www.ud.com/products/documentation/
17) Berkeley Open Infrastructure for Network Computing. http://boinc.berkeley.edu

(平成 17 年 1 月 25 日受付)
(平成 17 年 5 月 9 日採録)

蟻川　浩

1976 年生．1997 年長野工業高等専門学校機械工学科卒業．1999 年長岡技術科学大学工学部創造設計工学課程卒業．2004 年奈良先端科学技術大学院大学情報科学研究科博士後期課程研究指導認定退学．現在，関西大学経済・政治研究所政策グリッドコンピューティング実験センター特任研究員．グリッド環境構築およびシミュレーションツールのグリッド環境への適用に関する研究に従事．

増田　慎吾（学生会員）

2003 年滋賀大学教育学部卒業．2005 年奈良先端科学技術大学院大学情報科学研究科博士前期課程修了．現在，同大学博士後期課程在学中．グリッド環境におけるワークフローシステムに関する研究に従事．IEEE 会員．

古田　忠臣

1974 年生．1997 年神戸大学発達科学部卒業．1999 年神戸大学大学院総合人間科学研究科人間環境科学専攻修士課程修了．2003 年神戸大学大学院自然科学研究科構造科学専攻博士課程修了．2003 年 10 月より神戸大学理学部産学連携研究員．2004 年 4 月より神戸大学理学部学術研究員．2005 年 4 月より東京大学分子細胞生物学研究所にて科学技術振興機構 CREST 研究員．理学博士．タンパク質の立体構造予測，ダイナミクスに関する研究に従事．日本物理学会，日本生物物理学会各会員．

金　文珍

2004 年神戸大学大学院自然科学研究科分子集合科学専攻博士課程修了．2004 年 4 月から 11 月まで JST 特別研究員，2004 年 12 月より神戸大学理学部．理学博士．

朴　聖俊（正会員）

1998 年専修大学経営学部卒業．2005 年東京工業大学大学院総合理工学研究科知能システム科学専攻修了．博士（工学）．2005 年 4 月より神戸大学理学部学術研究員，現在に至る．進化型計算，バイオインフォマティクス，立体構造予測の研究に従事．人工知能学会，日本バイオインフォマティクス学会，日本分子生物学会各会員．

高田 彰二

1988年京都大学理学部卒業．1990年京都大学大学院理学研究科化学専攻修士課程修了．1991年岡崎国立共同研究機構技官，1995年日本学術振興会研究員，1998年神戸大学理学部講師を経て，2001年より同大学助教授，現在に至る．理学博士．生物物理，理論的タンパク質構造と機能解析，タンパク質立体構造予測の研究に従事．

藤川 和利（正会員）

1988年大阪大学基礎工学部情報工学科卒業．1991年大阪大学大学院基礎工学研究科博士後期課程退学後，同年大阪大学基礎工学部助手等を経て，2002年奈良先端科学技術大学院大学情報科学センター助教授，2005年同大学情報科学研究科助教授，現在に至る．博士（工学）．分散処理システム，マルチメディアシステムの研究開発に従事．電子情報通信学会，IEEE，ACM各会員．

砂原 秀樹（正会員）

1983年慶應義塾大学工学部電気工学科卒業．1989年慶應義塾大学大学院博士課程修了．1988年電気通信大学情報工学科助手，1994年奈良先端科学技術大学院大学情報科学センター助教授を経て，現在，同大学情報科学研究科教授．工学博士．インターネット，モバイル/ユビキタスコンピューティング，大規模広域分散環境，並列処理，オペレーティングシステム，電子図書館に関する研究に従事．電子情報通信学会，ソフトウェア科学会，Internet Society，ACM，IEEE各会員．