iアプリを用いた数値計算の可能性

髙田雅美[†,☆] 柴山智子[††] 渡辺知恵美[†]
庄野逸[†††] 城和貴[†]

携帯電話会社 NTT Docomo の提供する iアプリは，携帯端末において，様々な Java アプリケーションを実行することが可能である．iアプリ対応の携帯端末は，計算機としては貧弱ではあるが，ネットワーク機能・計算リソースを有しており，2003 年 11 月の時点では，その市場稼動台数は，2,000 万台を突破している．本研究では，アイドル状態にある携帯端末において，iアプリを用いた数値計算を行うための分散処理システムによって，計算リソースを有効活用する方法を提案する．個々の携帯端末は，貧弱な計算リソースしか持たないため，処理すべき計算問題は，分散化される必要がある．そこで，我々は，この分散処理のための 1 つのアプローチとして分散遺伝的アルゴリズムを iアプリによって実装し，計算問題解決のためのプロトタイプを作成した．

The Possibility for Numerical Calculations with i-APPLI

Masami Takata,[†,☆] Tomoko Shibayama,[††] Chiemi Watanabe,[†]
Hayaru Shouno[†††] and Kazuki Joe[†]

The i-APPLI, provided by NTT Docomo Co.ltd, makes Java^{TM} applications execute on mobile phones. Although a mobile phone has just poor calculation resource, it provides data communication facility as well as calculation resource with us. Namely, a mobile phone can be a node of a distributed processing system. In November 2003, over 20 millions mobile phones with i-APPLI have been shipped to the market, and most of them are considered as an idle state. A computational resource consisting of many idle mobile phones is attractive for a distributed processing system. In this paper, we propose a distributed processing system using mobile phones for numerical calculations. We implement a prototype, that is a distributed genetic algorithm using i-APPLI, as one of approaches for numerical calculations with mobile phones.

1. はじめに

情報化社会が高度化するにつれ，エンドユーザが用いる情報端末はダウンサイジングされる傾向にある．現在，主に使われている情報端末としては，計算および通信を行うためのリソースを有したパソコン・PDA・携帯端末などがあげられる．これらの情報端末のうち，パソコンや PDA は，比較的複雑な計算が可能な反面，典型的なエンドユーザにとって，アプリケーションの入手・設定・操作は複雑なものとなっている．それに比べて，携帯端末は，処理性能が劣るもののアプリケーションの入手方法などが単純化されている．そのため，幅広い世代の人々に情報端末として利用されている．

携帯電話会社 NTT Docomo[8] が提供する携帯端末では，通信機能として，通常の音声通話を行う電話としての機能と，ネットワーク接続を行う iモード[6] との 2 種類がある．本論文で研究対象とした iアプリ[4] は，この iモードを用いてサーバから実行可能な Java^{TM} プログラムをダウンロードし，携帯端末本体で実行させるという形をとる．iアプリ対応の携帯端末は，2001 年の *503i* シリーズ発表以来，急速に市民権を得て，すでに 2,000 万台以上が市場に流通している[5]．

iアプリ対応の携帯端末は，計算機としては貧弱であるが，ネットワーク機能とプログラミングツールを用いてユーザが自由に使える計算機としてのリソースを有している．この計算機としてのリソースを利用す

† 奈良女子大学大学院人間文化研究科
Graduate School of Human Culture, Nara Women's University
†† 日本総合研究所
The Japan Research Institute Limited
††† 山口大学工学部
Faculty of Engineering, Yamaguchi University
☆ 現在，独立行政法人科学技術振興機構さきがけ（京都大学大学院情報学研究科）
Presently with PRESTO, JST (Graduate School of Informatics, Kyoto University)

る様々なiアプリが開発されているが，実際に利用されているものは，画像・音声の再生，ゲーム，占いなどのエンタテイメント性の強いものと，時刻表，スケジュール帳，株価情報，地図・渋滞情報の表示などの社会的実用性の強いものが主である．すなわち，どちらも計算を行う主体としてのアプリケーションではない．また，待ち受け画面用のiアプリ以外は，常時実行を必要としないため，現在流通しているほとんどすべての携帯端末でiアプリ用の計算リソースが余っている状態であると考えられる．そこで，本研究では，iアプリを用いた分散処理を行うことによって，アイドル状態にある圧倒的多数の携帯端末のリソースを利用した大規模数値計算の可能性を検討することを目的とする．

携帯端末で大規模数値計算を行うためには，分散処理を行う方法が考えられる．この際，十分な数の携帯端末数を確保しなければならない．なぜならば，パソコン1台分の能力を得るためには，数百台の携帯端末を必要とするからである．これらのユーザを確保するためには，分散処理を待ち受けiアプリのバックグラウンド処理として行うコンテンツなどの提供や課金システムの確立などを行うことで対処可能であると考えられるが，本論文の研究対象範囲外である．

本論文では，数値計算の第1歩として，遺伝的アルゴリズム（Genetic Algorithm: GA）[3] の手法の1つである分散GAのうち島モデル[9] を用いたものを *503i*，*504i* シリーズに実装し，携帯端末における性能評価を行う．

以下，2章において，iアプリの特徴について簡単に述べる．iアプリを用いた数値計算の可能性を検証するための1つのアプローチとして，iアプリを用いて島モデルのGAが実行可能であるかどうか3章において検討し，ナップサック問題を解くことにより性能評価を行う．

2. 携帯端末の性能と分散処理の可能性

iアプリは2種類に分類することができる[1]．1つは，非接続型であるスタンドアロンタイプである．このタイプのiアプリは，ネットワーク通信を必要とせず，携帯端末のみで動作するアプリケーションであり，パケット通信は，最初のiアプリダウンロードのときだけ行われる．もう1つは，データ更新にネットワーク接続を必要とするサーバ連動タイプである．このタイプは，サーバとの通信を介して，他の携帯端末やサーバとのデータ通信が可能である．つまり，サーバを経由することで，複数の携帯端末においてデータ共有が可能となる．この際の接続方法には，ユーザの手動による方法と，iアプリ自体が一定時間間隔で自動的に行う方法がある．ただし，データを通信する場合以外，ネットワークに携帯端末が接続している必要性はない．また，将来的には，Bluetooth^{TM} などが普及すれば，サーバを介さずに携帯端末どうしのみで情報交換を行える可能性があるが，大規模計算を携帯端末のような非常に小さい計算リソース上で実装するためには，データはどこかで一元管理されているシステムのほうが望ましいと考えられるため，サーバを介してデータ共有を行うほうが現実的である．

データ通信やiアプリ自体の実装には，Java^{TM} 言語が用いられている．そのため，一般的なプログラマにとって，多様なアプリケーションを容易に開発することが可能である．ただし，省電力およびメモリなどの制約条件が厳しい機器向けのJ2ME規格[12] に従ったプログラムを記述しなければならないので，これらの制約条件には留意する必要がある．

携帯端末の種類によってプログラムサイズや使用可能なファイルタイプの制限があるものの，シリーズが進むにつれ，**表1** に表されているように，標準的なスペックは向上している．**表2** は，市販されている数種類の携帯端末上で，付録A.1のiアプリを実行し，MIPS値を測定した結果である．得られたMIPS値は，各社ばらつきがあるものの，*504i* シリーズ以降の機種は数MIPSといえるだろう．これは，1980年代に使われたDEC社のVAX-11/780シリーズやSun Microsystems社の初期のSPARCプロセッサに相当するものである．

また，プログラムの実装において，記憶領域としてヒープが必要となる．**表3** は，各携帯端末シリーズのヒープ容量を表す．ただし，ヒープ容量が，Javaの

表1 *503i*，*504i*，*FOMA*，*505i* の標準スペック（KB）
Table 1 Spec of *503i*, *504i*, *FOMA*, and *505i* (KB).

	503i	*504i*	*FOMA*	*505i*
通信サイズ（上り）	5	5	5	10
（下り）	10	10	10	20
.Jar 容量	10	30	30	30
Scratch Pad 容量	10	100	200	200

表2 各携帯端末における MIPS 値
Table 2 MIPS values of sevelal mobile phones.

	503i	*503is*	*504i*	*504is*	*505i*
Fujitsu (F)	0.11		2.56	2.56	
Panasonic (P)			0.92	0.93	
NEC (N)	0.21	0.21	10.0	10.0	
MITSUBISHI (D)			0.47	-	7.35
Sony (SO)	0.60	1.11	1.11	-	1.48

オブジェクトなどを格納する Java ヒープと，メディアデータや画面表示のためのリソースを格納するネイティブデータヒープの区別がある場合は「Java ヒープ/ネイティブデータヒープ」の順序で記載し，区別がない場合は Java ヒープ容量を記載している．

表 3 に示されているように，携帯端末のヒープ領域は小さく，大きな計算データを格納しておくことは困難である．ゆえに，携帯端末を対象として，実用的な数値計算を処理させるためには，数万台の携帯端末にサーバ連動タイプの i アプリを実装した Master-Slave 型の分散処理システムの構築が必要となる．

一般的な分散処理システムは，単体の計算機で処理するには過大な計算量やデータ量を持つアプリケーションを，ネットワークに接続された複数の計算機を利用して実行する．ただし，この場合の計算機とは，一般的に使用されているパソコン，ワークステーション，もしくはクラスタ型コンピュータを構成するユニットであり，携帯端末よりもはるかに高い計算能力を持つ反面，その構成台数は数 100 台程度のオーダで考えられている．

一方，i アプリ対応携帯端末は，低性能ながらも，普及台数は 2,000 万台以上と非常に多く，それらのうち，一部分しか参加者がいなかったとしても，数万台の携帯端末を分散処理のユニットとして確保できる可能性がある．また，携帯端末ユーザは，普段使用していない余剰計算リソースのみを提供するだけなので，比較的負担が少ないと考えられる．この余剰計算リソースを利用し，携帯端末間で分散処理することによって，様々な研究のための数値計算を行うことが可能であると考えられる．すでに，パソコン上では，この余剰計算リソースを用いて解探索を行うというコンセプトは，実現されており，電波望遠鏡で得られたデータをスクリーンセーバのバックグラウンド処理で解析するための SETI@home [11] や，タンパク質の構造解析に用いられる Folding@home [2] などがよく知られている．

本研究では，貧弱な計算能力しか有さない携帯端末を利用して実用的な数値計算を可能とするための分散処理の枠組みを提案する．ただし，データの一元管理のためにサーバを介した通信が必要となるため，Master-Slave 型の分散処理システムを構築するものとする．このプロトタイプとして，GA の一種である島モデルを i アプリ用に変更し，その有効性を検証する．

表 3 ヒープ容量（KB）
Table 3 Heap for i-APPLI (KB).

	Fujitsu (F)	Panasonic (P)	NEC (N)
503i	600	288/150	96/286
503is	600	288/150	96/286
504i	1000	1500	544/600
504is	1500	1500	544/1030
2051	600/700	-	760/1228
2102V	1536/1024	760/1228	760/1488
505i	2000/1500	3500	1494/1472
	MITSUBISHI (D)	Sony (SO)	Sharp (SH)
503i	1110/220	350	-
503is	1110/220	350	-
504i	1024/524	1000	-
505i	1536/2048	1800	1700

3. i アプリを用いた島モデルの GA

i アプリとして GA を実装する場合，機種固有のマルチメディア系 API を用いる必要がないため，機種別に i アプリを実装しなくてもよい．ただし，i アプリは小型デバイス用の Java である J2ME 規格に従っているため，浮動小数点演算を用いることができない．すなわち，整数演算のみを用いてアルゴリズムを構築しなければならない．また，携帯端末の場合，一般的な計算機と比較して，ヒープ容量や演算速度が著しく劣るため，GA の実行時にとりうる個体数に関しては，大きな制限を受ける．このような制限のために，1 つの携帯端末で完結した GA を行うよりは，複数の携帯端末で島モデルの GA [9] と呼ばれる分散協調型の GA を実装したほうが，実装上有利であると考えられる [10]．島モデルの GA は，母集団を複数の部分集団（島）に分割し，島ごとに遺伝的操作を適用する手法である．また，一定の世代更新が終了するごとに，他の島と個体を交換する．この操作は移住（migration）と呼ばれる．島モデルの GA は，通常の単一母集団として実装した GA と比較して，より適合度の高い優良解を発見することが可能である．

本研究では，一例としてナップサック問題を解くために，i アプリを用いた島モデルの GA を実装し，結果の評価を行う．

以下，3.1 節において，GA を i アプリで実装するための工夫について述べる．次に，GA を用いたナップサック問題について，3.2 節で説明し，3.3 節において，その実験結果を示す．

3.1 i アプリ上への島モデルの GA の実装

一般に，GA がその真価を発揮するのは，遺伝子情報が非常に多い場合で，他の探索手法では適切な遺伝子を決定することが困難な場合が多い．この場合，遺伝子長が長くなり，優良解を得るためには GA の各世代における個体数を多くすることが望ましい．i アプリにおける実装において，各個体の遺伝子情報は携帯端末上のヒープ領域に保持される．しかしながら，表 3

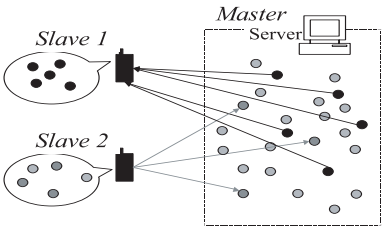


図 1 iアプリ用の島モデルの GA の概念図
Fig. 1 The model of island-GA with i-APPLI.

に示されているように，現状の携帯端末のヒープ容量は小さいため，十分な個体数を持つ母集団を保持することは難しい．また，演算速度もそれほど期待できないという理由から，島モデルの GA[9] を採用した．

島モデルの GA を実装する場合，移住を行う必要があるが，各島単位である携帯端末間では，現在のところ，直接通信を行うことは難しい．IrDA などの赤外線通信や Bluetooth^{TM} などを使用することも考えられるが，これらの機能が存在しない機種もあるため，あまり実用的ではない．より遠くの携帯端末と情報交換を行いたければ，何らかの通信サーバを介するほうが現実的である．そこで，我々のiアプリを用いた島モデルの GA の実装においては，Master-Slave 型の分散処理を用いることが可能となるように，島モデルの GA を変更することとした．

我々の用いた Master-Slave 型の島モデルの GA は，**図 1** のように表される．この際，母集団は Master であるサーバ内に置かれる．Slave である各携帯端末では，母集団から取り出された部分集団を用いて，世代更新を行う．各携帯端末で得られた更新結果は，サーバに送信され，サーバでは，更新結果をもとに，母集団の更新を行う．サーバと携帯端末間においては通信を行う必要があるが，この通信には HTTP プロトコルを用いた．iアプリ対応の携帯端末は，DoJa[4] と呼ばれる J2ME から派生した通信機器用の Java が実装されており，DoJa には HTTP 通信のサブセットがライブラリとして提供されている．Web サーバをデータ共有のインタフェースとする利点は，iモードとの親和性だけでなく，新たにクラスライブラリを構築する必要がなく，実行時に少しでも多くのヒープを利用できることもあげられる．本実装方式では，通信の核となる部分は HTTP プロトコルにおける GET 要求および POST 要求によって実装した．

本研究では，改良方法として，サーバに母集団を置き，携帯端末では，母集団から取り出した部分集団を用いて世代更新を行う．

このiアプリのために改良されたアルゴリズムは，

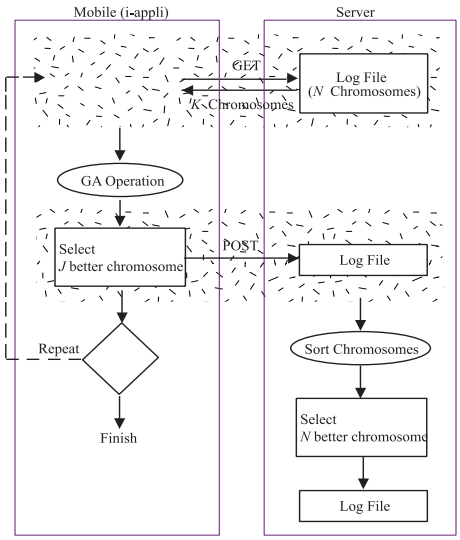


図 2 iアプリで GA を実行するためのアルゴリズム
Fig. 2 An algorithm for GA with i-APPLI.

図 2 のようになる．

(1) （サーバ）N 個からなる母集団を生成し，ログファイルに保存．
(2) （携帯端末）iアプリを起動．
(3) （携帯端末→サーバ）GET 要求による個体取得リクエストの発行．
(4) （サーバ）母集団から $K(1 < K \leq N)$ 個の個体をランダムに選択．
(5) （サーバ→携帯端末）サーバから携帯端末に，選択された K 個の個体情報を通信．
(6) （携帯端末）GA のオペレーション（選択，交叉，突然変異）を G 世代目が生成されるまで繰り返す．
(7) （携帯端末）優良個体を $J(1 \leq J \leq K)$ 個選択．
(8) （携帯端末→サーバ）POST 要求を用いて，携帯端末からサーバに，選択された J 個体を通信．
(9) （サーバ）受信した J 個体と母集団の N 個体をソート．
(10) （サーバ）ソートされた $N + J$ 個体から，適応度の高い N 個体を選択し，ログファイルに上書き保存．
(11) （携帯端末）iアプリを終了しない場合，手順 (3) に戻る．

手順 (3) から (5) は，各携帯端末からの GET 要求の呼び出しに応じて実行される．同様に，手順 (8) から (10) は，各携帯端末からの POST 要求の呼び出しに応じて実行される．今回の実験で用いた実装方式では，手順 (8) で用いられる POST 要求は，GA

の操作終了後，ユーザが手動で携帯端末モニタ上の送信メニューを選択することによって実行されるように設定した．これは，電波圏外でiアプリが終了した場合，サーバに情報を送信できないという問題を回避し，確実に実験結果を送信するための措置である．しかし，仮に携帯端末からサーバに優良個体情報が送信できなかったとしても，島に割り当てられた小集団の絶滅を意味するだけであり，母集団情報が変更されることはない．そのため，再びサーバの母集団情報から小集団を受け取ることによって，島である携帯端末においてGAの操作を行うことが可能となり，世代更新結果を再生成することが可能となる．ゆえに，今後の実用化においては，ユーザの手間を減らすために，このPOST要求の呼び出しは自動的に行われるようにする予定である．

一般に，従来の島モデルのGA[9]では，ある島において個体がすべてある特定の局所解に陥ったとしても，他の島からの移住によって，局所解から脱する可能性があるという特徴を持っている．改良された島モデルのGAにおいて，同様の性質を持たせるために，サーバ側で母集団を管理し，携帯端末ではその一部分を用いて世代更新を行うこととする．つまり，ある携帯端末において得られた結果であるJ個の個体は，局所解を示すものであったとしても，サーバ側の母集団において，一部分の個体として扱われる．さらに，次のiアプリ起動時に携帯端末に送信される部分集団は，母集団内からランダムに選択されたものであるため，局所解のみでなく，他の携帯端末から母集団に送信された個体も含まれる可能性が高い．このため，各携帯端末において，局所解からの脱出が可能となり，母集団に含まれるすべての個体が局所解となる可能性は低い．ただし，手順(10)において，適応度の高い個体を優先的に保存しているため，母集団の更新回数が増すほど，収束速度は緩やかではあるが，各個体の多様性がなくなる傾向が強くなる．ゆえに，母集団において，局所的な解に陥る危険性が皆無となるわけではない．

また，島モデルの性質より，携帯端末の結果が局所解であっても支障がないため，1つの携帯端末上のGA操作において，個体の多様性の維持を重視する必要はそれほどない．ここでは，K個の部分集団を携帯端末に割り当てているが，Kを数十個程度にすれば，十分実用的な長さの遺伝子を割り当てることが可能である．世代更新回数Gに関しては，携帯端末の計算資源である演算能力や消費される電力などを考慮した場合，長時間世代交代させることはそれほど得策ではない．また，終了世代において母集団に返される個体が未成熟であったとしても，それほど問題は起こらないと考えられる．ゆえに，世代更新回数Gを小さく設定しても支障はない．

3.2 ナップサック問題への適用

本研究では，数値計算の実行可能性を検証するために，ナップサック問題を解くためのGAを具体例として採用した．ナップサック問題とは，L個の荷物があり，それぞれに重さ$w_j(1 \leq j \leq L)$と価値c_jとが定義されているような状況において，ナップサックの耐荷重量をWと設定した場合に，どの荷物をナップサックに詰めれば最も価値を高くすることができるかということを判定する問題である．この問題は，NP完全問題であることが知られており，総当たり方式で真の解を得るために必要な時間は，荷物の個数Lの指数オーダに比例した時間となる．

GAを用いて実装するために，集団個体数をMとし，この中のi番目の個体の遺伝子を$\mathbf{s}_i$で表すものとする$(1 \leq i \leq M)$．荷物をナップサックに入れるか入れないかは，$1bit$で表現可能なので，i番目の遺伝子$\mathbf{s}_i$のj番目の要素を$s_{i,j}$で表し，0または1の2値をとるものとする．この場合，j番目の荷物を入れる場合は$s_{i,j}=1$，入れない場合は$s_{i,j}=0$として表すものとする．今，L個の荷物を考えているので，遺伝子の長さはLとなる．

i番目の個体の総重量W_iは，

$$W_i = \sum_{j=1}^{L} s_{i,j} w_j \tag{1}$$

とする．このときの総価値C_iは，

$$C_i = \sum_{j=1}^{L} s_{i,j} c_j \tag{2}$$

で表される．個体の適応度F_iは，総価値とナップサックの制約条件

$$W \geq W_i \tag{3}$$

を考え，

$$F_i = \max\{0, C_i - \alpha(\max\{0, W_i - W\})\} \tag{4}$$

とする[9]．ただし，αは，ペナルティパラメータである．仮に，i番目の個体の総重量がナップサックの耐荷重量を超過している場合，$W_i - W$が正となり，ペナルティとして，超過荷重とαの積を総価値から減じたものが適応度となる．ペナルティを強くする場合には，αを大きな値として適用すればよい．

GAの世代交代モデルには，単純GAを用いた．すなわち，親集団からの交配選択方法にルーレット選択を用い，生存選択には生成された子集団が次世代の親

集団となるモデルである．したがって，携帯端末上では世代交代によって優良解が破壊される可能性もある．しかし，各携帯端末の最終的な個体情報は，サーバ側に順序付けられたうえで保持され，なおかつ，携帯端末が返す個体数 J とサーバ側の母集団の個体数 N には $N \gg J$ という関係が成り立つ以上，1 つの携帯端末である一試行の結果が悪くてもサーバ側の母集団に与える影響はそれほど大きくないと考えられる．

親世代から子世代への移行にあたって，ルーレット選択による交叉と突然変異によって，新たに M 個体生成する．

交叉方法は，計算の簡単さを考え，一点交叉とする．この際，交叉地点は，ランダムに選択されるものとする．

各携帯端末においては，GA の最終世代の解を収束させる必要性はそれほどなく，むしろ，より幅広い探索が必要となると考えられる．そのため，ここでは，突然変異率を通常の単純 GA で行う場合よりも高めの 5% に設定した．突然変異率を高めに設定した場合，GA 的な要因よりもランダム探索的な要因が強くなることも考えられるため，予備実験としてパソコン上において，上記ナップサック問題の解のランダム探索を行った．この予備実験では，ランダム探索として，その一手法であるマルコフ連鎖モンテカルロ法（Markov Chain Monte Carlo: MCMC）を採用し，実験条件は $L = \alpha = 16$ とし，500 ステップを 1 試行とした．これは，GA において 50 個体で 10 世代の探索に相当すると考えられる．MCMC 法による解探索を 100 試行行った結果，準最適解（適応度 90 前後）には収束するものの，最適解（適応度 104）に収束することはなかった．また，ほぼすべての試行において，約 10 ステップで収束し，それ以降，準最適解から抜け出すことはなかった．

3.3 実行結果

3.2 節で説明した GA を携帯端末上で i アプリとして実装したものを用いて，実行結果が妥当なものであるか検証した．

使用した機種は，*SO504i*，*SO503is*，*N503i*，*P504i* である．携帯端末のシリーズが異なる場合，古いシリーズに合わせた i アプリを開発するか，各シリーズの性能を存分に利用するために各シリーズ専用の i アプリを開発しなければならない．本研究において，実用的な数値計算が実行可能であることを検証するにあたり，各シリーズの性能を考慮しなくてすむように，古いシリーズである *503i* に合わせた開発を行った．

実行にあたって，ナップサック問題における荷物の個数 L は 16 個とした．各荷物の価値 c_i は $\{8, 9, 10, 8, 9, 10, 7, 7, 6, 9, 8, 7, 6, 9, 10, 7\}$ とし，重量 w_i は $\{10, 6, 7, 7, 8, 9, 6, 9, 9, 10, 8, 7, 8, 7, 10, 9\}$ と設定した．また，ナップサックの耐重量は，$W = 100$ とする．この問題における最適解は，6 種類存在し，その適応度は 104 となる．

GA で用いられるパラメータは，母集団をサーバで保持するため，大規模な $N = 200$ とし，携帯端末で実行する小集団の個体数は $K = M = 50$，携帯端末からサーバに送られる個体数は $J = 5$ とした．ペナルティパラメータは遺伝子長と同じ $\alpha = 16$ とした．また，GA の終了世代数は，未発達な状態であっても支障がないため，$G = 5$ とした．

J2SE を用いて改良された GA を通常のパソコン上で実行する Java アプリケーションと i アプリをそれぞれ 10 回実行した結果が，**表 4** である．表 4 より，J2SE の場合も i アプリの場合も，最終世代における最大適応度の範囲は，ほぼ一致することが分かる．また，同じ最大適応度であっても，異なる遺伝子情報を持つ個体が結果として得られ，同一機種で実行した場合も，実行ごとに異なる最大適応度と遺伝子情報が得られた．つまり，実行ごとに異なる個体が生成されているものと考えられる．

図 3 は，i アプリの連続 5 回の繰返しを 10 セット行った結果得られた母集団の最大適応度の推移を表す．各回終了後，母集団に携帯端末から $J = 5$ 個の優良個体が移住することによって，図 3 に表されるように，

表 4 i アプリのための GA を用いた場合の J2SE と i アプリの 10 回分の最大適応度
Table 4 The maximum fitness values of GA with J2SE and i-APPLI.

J2SE	87	91	97	98	97
	93	95	96	96	98
i-APPLI	98	89	95	91	91
	83	85	97	97	92

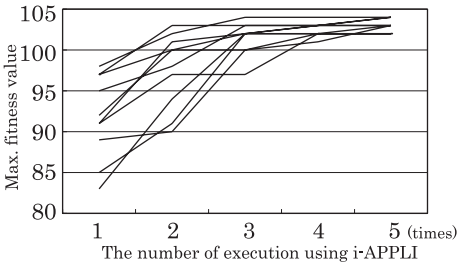


図 3 i アプリの連続実行による最大適応度の推移（$L = 16$）
Fig. 3 The maximum fitness values with i-APPLI ($L = 16$).

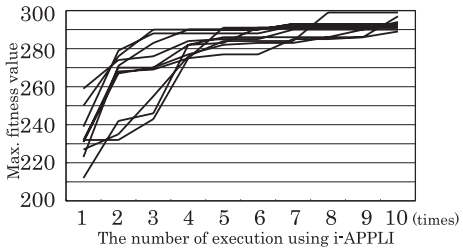


図 4 iアプリの連続実行による最大適応度の推移（$L = 50$）
Fig. 4 The maximum fitness values with i-APPLI ($L = 50$).

iアプリの実行回数にともなって，最大適応度が大きくなっていることが分かる．さらに，最良解である適応度 104 で遺伝子情報の異なる個体が生成される場合もあった．つまり，iアプリのために改良したアルゴリズムの手順 (8) から (10) で実行される移住によって，より優良な解を生成し，最良解を得ることが可能である．

iアプリの携帯端末上での実行時間を計測するために，*SO503is* を用いた．一般的な島モデルの GA を Master-Slave 型に改良したアルゴリズムを実装した iアプリを実行した場合，平均実行時間は，サーバから携帯端末への初期データ通信と携帯端末用への受信データのデコードなどの前処理に関して 14 秒，GA の実行に関して 3 秒，最終世代の結果の送信に関して 3 秒であった．前処理に関する時間のほうが，最終世代の結果の送信時間より長い．これは，最終世代の結果として送信される個体数が $J = 5$ であるのに対して，前処理では個体数 $K = 50$ が受信されるためであると考えられる．

次に，もっと大規模な遺伝子長を対象とする GA の実行が可能であるかどうか検証するために，以下の実験を行った．実行にあたって，ナップサック問題における荷物の個数 L は 50 個とし，各荷物の価値 c_i と重量 w_i は 6 から 10 の自然数を与えた．また，ナップサックの耐重量は，$W = 300$ とした．GA で用いられるパラメータは，先に行った実験と同じものを用いた．ただし，ペナルティパラメータは遺伝子長と同じ $\alpha = 50$ とした．

図 4 は，iアプリの連続 10 回の繰返しを 10 セット行った結果得られた母集団の最大適応度の推移を表す．図 4 に表されるように，$L = 16$ の場合同様，iアプリの実行回数にともなって，最大適応度が大きくなっていることが分かる．また，iアプリの実行回数が 5 回目までは，実行回数にともなって，最大適応度が大きくなり，6 回目以降は，ほぼ横ばい状態ではあるものの，若干最大適応度が大きくなっている．これは，移住の効果が効いているものと思われる．

以上より，島 GA を iアプリに実装する際，改良されたアルゴリズムは，遜色のない解を探索しうることが判明した．また，最終的な解を得るためには何度も実行しなければならないが，母集団を大きくとることが可能であるため，探索空間を広げることは容易である．つまり，現在のヒープ容量などの携帯端末性能を考慮すると，大規模探索をするためには，実行回数は多くなるが，改良されたアルゴリズムを用いることが有効であると考えられる．

4. ま と め

2003 年 11 月の時点では，携帯通信の市場には 2,000 万台以上の iアプリ対応の携帯端末が普及しており，この携帯端末は，ネットワーク機能と計算リソースを有した小型計算機と見なすことができる．本論文では，iアプリ対応の携帯端末の余剰計算リソースを利用した数値計算の可能性について述べた．その一例として，iアプリを用いて島モデルの遺伝的アルゴリズム（GA）を実装した．実装にあたって，Master-Slave 型に島モデルの GA を改良した．この改良されたアルゴリズムを検証するために，ナップサック問題を解いた．その結果，ナップサック問題のスケールに関して問題があるものの，一般的な島モデルの GA の実行と比較して，ほぼ遜色のない性能結果を示せた．2003 年 6 月より発売が開始された *505i* シリーズでは，各機種とも，ヒープ容量，ScratchPad 容量ともに最大容量が 2 倍以上に増加されたため，より大規模な数値計算が可能になるものと思われる．

今後は，iアプリによる数値計算コンポーネントの開発を行うとともに，実用化に対応するために，携帯端末の待ち受け画面型 iアプリとしての開発を行う予定である．

参 考 文 献

1) アスキー書籍編集部：iモード Java プログラミング，ASCII (2001).
2) Folding@home スタンフォード大学. http://folding.stanford.edu/
3) Goldberg, D.E.: *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison Wesley (1989).
4) iアプリコンテンツの作成について. http://www.nttdocomo.co.jp/p_s/imode/java/
5) iモード契約数. http://www.nttdocomo.co.jp/corporate/

report/contract/index.html
6) iモード製品サービス.
http://www.nttdocomo.co.jp/p_s/imode/
7) MIPS 値計算用ベンチマーク.
http://alice.ics.nara-wu.ac.jp/ takata/i
8) NTT 公式サイト.
http://www.nttdocomo.co.jp/
9) 三宮信夫，喜多　一，玉置　久，岩本貴司：遺伝アルゴリズムと最適化，朝倉書店 (1998).
10) Tanese, R.: Distributed Genetic Algorithms, *Proc. 3rd Intl. Conf. on Genetic Algorithms*, pp.434–439 (1989).
11) The Search for Extraterrestrial Intelligence.
http://www.planetary.or.jp/setiathome/home_japanese.html
12) Sun J2ME 規格.
http://java.sun.com/j2me

付　　録

A.1 MIPS 値を計るための i アプリ用プログラム

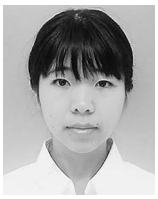
MIPS 値を測定するにあたり，ベンチマーク[7] を作成した．作成にあたり，文献 1) のスタンドアロン・アプリケーションで紹介されたユーティリティiBenchを利用することとした．利用クラスは，iアプリ起動時の設定に関するクラス（iBench.java）と時間測定に関するクラス（TimeSpan.java）である．

終了画面には，1,000,000 ループ内において，足し算を 1 回実行した場合の実行時間を 1 行目に，足し算を 2 回実行した場合の実行時間を 2 行目に表示する．この際，実行時間の単位は，ミリ秒とする．また，3 行目には，2 行目の実行時間から 1 行目の実行時間を引いた値を出力する．この実行は以下のような MIPS 用のクラスを作成することによって行う．

```
// MIPS 用プログラム
import com.nttdocomo.ui.*;

public class MIPS extends Canvas{
  boolean endFlag = false ;
  TimeSpan spn1,spn2;
  int width, height;

  MIPS(){
    spn1 = new TimeSpan();
    spn2 = new TimeSpan();
    width = Display.getWidth();
    height = Display.getHeight();
  }

  public void paint(Graphics g){
    long time,time1, time2;
    String str;
    if(endFlag == false){
      g.lock();
      g.clearRect(0,0,width,height);
      g.drawString("MIPS Start", 0, 20);
      g.unlock(true);
// ループ内の足し算 1 回の測定
      spn1.start(); // 測定開始
      mips_loop1(1000000); // 実行
      spn1.stop(); // 測定終了
// ループ内の足し算 2 回の測定
      spn2.start();
      mips_loop2(1000000);
      spn2.stop();
      endFlag = true;
      repaint();
    } else { // 結果表示
      g.lock();
      g.clearRect(0, 0, width, height);
      g.drawString("MIPS End", 5,20);
// 実行時間の取得
      time1 = spn1.getResult();
      time2 = spn2.getResult();
      time = time2 - time1;

      str = time+"msec";
      g.drawString(str,5,80);
      g.drawString(spn1.toString(),5,40);
      g.drawString(spn2.toString(),5,60);
      g.unlock(true);
      endFlag = false;
    }
  }

// ループ内の足し算 1 回の測定
  public void mips_loop1(int size){
    int i;
    int itmp;
    for(i = 0; i <= size; i++){
      itmp = 1+1;
    }
  }
```

```
// ループ内の足し算 2 回の測定
  public void mips_loop2(int size){
    int i;
    int itmp;
    for(i = 0; i <= size; i++){
      itmp = 1+1;
      itmp = 1+1;
    }
  }
}
```

(平成 15 年 8 月 22 日受付)
(平成 15 年 11 月 14 日再受付)
(平成 15 年 12 月 20 日採録)



高田 雅美（正会員）

昭和 52 年生．平成 13 年奈良女子大学大学院人間文化研究科情報科学専攻修士課程修了．平成 16 年奈良女子大学大学院人間文化研究科複合領域科学専攻修了．博士（理学）を同大学より取得．平成 16 年独立行政法人科学技術振興機構戦略的創造事業の委嘱研究員として，京都大学大学院情報学研究科数理工学専攻数理解析分野にて従事．分散メモリ環境を対象とする並列プログラムの開発に関する研究に従事．

柴山 智子

昭和 52 年生．平成 15 年奈良女子大学大学院人間文化研究科情報科学専攻修士課程修了．同年日本総合研究所に入社．Java 言語を用いたシステムの開発に従事．

渡辺知恵美（正会員）

昭和 50 年生．平成 15 年お茶の水女子大学大学院人間文化研究科複合領域科学専攻博士後期課程修了．博士（理学）．平成 15 年 4 月より奈良女子大学大学院人間文化研究科助手．データ視覚化，VR システム，インタラクション，およびそれらに対する効果的なデータベース支援に関して興味を持つ．日本バーチャルリアリティ学会，日本データベース学会，ACM，IEEE-CS 各会員．

庄野 逸（正会員）

昭和 43 年生．平成 4 年大阪大学大学院基礎工学研究科物理系専攻生物工学分野修了．同年大阪大学基礎工学部助手．平成 9 年大阪大学大学院基礎工学研究科助手．平成 13 年奈良女子大学大学院人間文化研究科助手．平成 14 年より山口大学工学部助教授．神経回路モデル等の確率的情報処理に関する研究に従事．平成 11 年博士（工学）を大阪大学より取得．日本神経回路学会，電子情報通信学会，物理学会各会員．

城 和貴（正会員）

大阪大学理学部数学科卒業．日本 DEC，ATR 視聴覚研究所（日本 DEC より出向），(株）クボタ・コンピュータ事業推進室で勤務．平成 5 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入学．平成 8 年博士（工学）を同大学院大学より取得．平成 8 年同大学院大学情報科学研究科助手．平成 9 年和歌山大学システム工学部情報通信システム学科講師．平成 10 年同学科助教授．平成 11 年奈良女子大学理学部情報科学科教授．画像処理，文字認識，ニューラルネットワーク，並列計算機アーキテクチャ，自動並列化コンパイラ，並列計算機の解析モデル，視覚化等の研究に従事．IEEE，ACM 各会員．