

並列プログラミングの学習における教育用並列プログラミング言語の適用

水谷 泰治[†] 西口 敏司[†] 橋本 渉[†]

大阪工業大学 情報科学部[†]

1 はじめに

マルチコアプロセッサやGPU, PC クラスタなどが普及し, 並列計算が可能な計算環境が身近になってきている. これらの環境で並列プログラムを記述する場合, 環境毎に独特な書き方を要するプログラミング言語やライブラリを用いなければならない. これは並列プログラミングの学習を難しくする要因の1つとなっている. この問題に対処するために著者らは環境毎の独特な書き方を排した教育用の並列プログラミング言語を提案している[1, 2]. 本稿ではこの言語を用いて並列プログラミングの演習を行い, アンケートを行った結果について報告する.

2 教育用並列プログラミング言語

本研究で用いた教育用並列プログラミング言語[1] (T 言語と表記) について説明する. 並列プログラムは実行環境によって並列化の考え方や使用する言語やライブラリが異なる. 例えば, PC クラスタでは C/Fortran と MPI[3], GPU では CUDA[4], マルチコアプロセッサでは Pthreads 等が用いられ, 分散/共有メモリ型の考え方やプログラムの書き方, ライブラリの使い方は異なる. 並列プログラミングの初学者にとって並列化の考え方と同時に環境毎の独特な書き方を身に付けることは学習の負荷を高める. これに対し, T 言語では各環境の独特な書き方を簡略化した操作関数を用いて並列プログラムを記述する.

図 1 に T 言語で記述したプログラムとその実行のイメージを示す. 図 1 左のプログラムでは, main 関数において T_Run 関数を用いて関数 f を 4 つ並列に呼び出している. T 言語プログラムは tccrun コマンドによりコンパイルと実行を行う. 図 1 右は左のプログラムの実行状況を表す. 並列に呼び出された関数の中では, T_GetMyNum 関数によって関数呼び出しの各実体に対して一意な識別番号を取得できる. T_Send 関数と T_Recv 関数は第 1 引数でそれぞれ送信先と送信元の識別番号を指定し, 第 2 引数以降に送信・受信バッファの先頭アドレスとサイズ(bytes)を指定する.

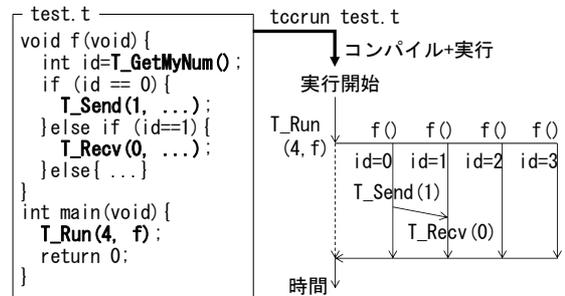


図 1. T 言語のプログラムと実行のイメージ

これらの操作関数は MPI や CUDA, Pthread などの各環境から抽出・簡略化したものであり, 各環境の煩雑さの軽減を意図している. 例えば, MPI の送受信関数では, 通信相手の識別番号だけでなく, 受信データを選別するためのタグや, 識別番号のグループを表すコミュニケータ, 受信状態を格納するステータス等を引数に指定するが, T 言語ではそれらを排除している.

3 演習内容

本演習は 2017/4/1(土)から 2017/6/17(土)までの期間に, 並列処理を専門分野とする研究室(著者の研究室)に配属された大学 4 年生 8 名を対象に実施した. この 8 名は C や Java によるプログラミングは学習しているが, 並列プログラミングの学習は本演習が初めてである.

今回使用した T 言語の処理系では, 4 コア CPU を搭載した PC 6 台を Gigabit Ethernet で接続した PC クラスタで並列プログラムを実行する. また, この処理系は実行時に通信の履歴を記録しており, 図 1 右のイメージ図に類似した図によって通信履歴を可視化する機能をもつ[1].

表 1 に課題の一覧を示す. 課題 1 から 4 は T 言語で実装し, 課題 5 は並列計算の分野で広く用いられている MPI と CUDA を用いて実装する. この意図は, 第 1 に, 最初は複雑な処理を排した T 言語を用いることで並列プログラミングの学習負荷を下げ, 並列化の考え方の学習に注力しても

表 1. 課題の一覧

課題	期間	概要	言語
1	1週	簡単な通信プログラム	T言語
2	2週	行列積(データ分散無し)	T言語
3	2週	行列積(データ分散有り)	T言語
4	3週	(選択) フラクタル画像生成 ガウスの消去法	T言語
5	2週	課題1-4をMPIで再実装 簡単なGPUプログラム	C+MPI CUDA

Parallel Programming Exercise with Educational Parallel Programming Language
Yasuharu Mizutani[†], Satoshi Nishiguchi[†], Wataru Hashimoto[†]
Faculty of Information Science and Technology, Osaka Institute of Technology[†]

表 2. 質問項目の一覧

1	T言語における並列プログラミングにおいて、T.Run関数を用いた関数の実体が複数呼び出されることで並列処理を行うことの理解の難易度はどうでしたか？
2	T.Run関数で呼び出された各関数はそれぞれが独立に動作しており、それぞれ個別の識別番号が割り振られているという概念の理解の難易度はどうでしたか？
3	T.Send関数とT.Recv関数の理解の難易度はどうでしたか？
4	tccrunコマンドの使い方の難易度はどうでしたか？
5	T言語プログラムの可視化の機能は使用しましたか？
6	MPIやCUDAを用いるよりもT言語を用いることで並列プログラミングの学習のハードルは下がったと思いますか？
7	課題5でMPIとCUDAを使ったプログラムを作成する際、課題4までで用いたT言語で役に立ったことや役に立たなかったことがあれば教えてください。
8	もし、T言語を用いずに最初からMPIまたはCUDAを用いて同じ課題の並列プログラミング演習を行った場合、全体的な学習負担はどうなると思いますか？
9	総合的に考えて、課題1から4までをT言語で学び、最後にMPIやCUDAを学ぶというスタイルは良かったと思いますか？

らう点である。第2に、高性能計算の分野で実際に用いられている環境にも慣れてもらう点である。受講生にはこれらの意図を伝えた上で演習に取り組んでもらった。

4 アンケート

全ての課題を終了した後、受講生に対して表2に示すアンケートを記名式で実施した。表2の各質問に対する選択肢を表3に示す。選択肢のある質問には理由を記述する欄も設けている。表4にアンケートの結果を示す。

質問1-4は多数が「簡単だった」と回答した。その理由として、T言語の送受信関数にタグが無い、操作関数の引数が少ない、といったことが挙げられた。一方、「どちらとも言えない」、「難しかった」と回答した理由には、初めての並列プログラミングであり並列に動作するイメージがつかみにくかった、初めは難しかった、といったものがあつた。

質問5より、通信状況を表す可視化機能はほとんど使われていないことがわかった。理由として、必要な場面がなかった、見方がわからなかった、といったものが挙げられた。

質問6と8は全受講者が「思う」「T言語よりも苦勞すると思う」と回答した。その理由として、T言語は引数が単純であり取っ掛かりやすかった、MPIは手続きが多く初学時には難しい、MPIやCUDAに比べて関数等が直感的でわかりやすい(以上質問6)、初めからMPIを用いれば慣れるまでの苦勞が大きい、必要な処理や引数が多く考えることが多くなり理解に時間がかかると思う(以上質問8)といったものがあつた。

質問7は、基本的な考え方を理解できていたためMPIやCUDAの特有の処理も理解しやすかった、引数は全然違うが考え方はT言語と同じなので役

表 3. 各質問の回答の選択肢

1-4	(1)簡単だった (2)どちらとも言えない (3)難しかった
5	(1)よく(4回程度以上)使用した (2)少し(1-3回程度)使用した (3)全く使用しなかった
6	(1)思う (2)どちらとも言えない (3)思わない
7	選択肢は無し
8	(1)T言語よりも苦勞すると思う (2)T言語とそれほど変わらないと思う (3)T言語よりも簡単に取っ組みせると思う
9	(1)良かった (2)どちらとも言えない (3)良くなかった

表 4. アンケート結果

		質問								
		1	2	3	4	5	6	8	9	
選 択 肢	(1)	5	7	7	8	0	8	8	6	
	(2)	2	0	0	0	2	0	0	2	
	(3)	1	1	1	0	6	0	0	0	

に立った、といった肯定的な回答であつた。

質問9では6名が「良かった」と回答した。理由として、T言語からMPIの扱いを類推できるためMPIを簡単に理解できる、T言語というローカルな言語だけでなく実際に使われているものに触れることができる、といったものが挙げられた。一方、「どちらとも言えない」と回答した2人の理由は、いずれもMPIやCUDAをもう少し触ってみたいかつたというものであつた。

以上のように、質問5を除き、ほとんどが肯定的な回答であつた。これらの回答結果と理由により、3節で述べた教育用言語を用いた演習の進め方によって並列プログラミングの初学時の負担を軽減できていると考えることができる。

5 おわりに

本稿では並列プログラミングの初学者に対して教育用並列プログラミング言語を用いたプログラミング演習を実施した。アンケート結果より、教育用言語を用いることで並列プログラミングの学習負担の軽減に一定の効果はあつたものと考えられる。今後の課題として、より多くの学習者に適用し、並列プログラミングの理解度まで考慮して検証することが挙げられる。また、可視化機能の改善も必要であると考えられる。

謝辞

本研究はJSPS科研費JP15K21511, JP15K12008, JP17K00500の助成を受けたものである。

参考文献

- [1] 田中寛章, 藤井健太, 磯淵郁也, 水谷泰治. "複数のハードウェアでの共通操作に着目した教育用並列プログラミング言語の提案". 第78回情報処理学会全国大会, 5ZC-02, (2016-03).
- [2] 田中寛章, 水谷泰治. "教育用並列プログラミング言語におけるCPU制御による見かけ上の速度向上率の増加". 第79回情報処理学会全国大会, 3ZC-08, (2017-03).
- [3] <http://www.mpi-forum.org/>
- [4] <https://developer.nvidia.com/cuda-zone/>