マルチコアクラスタ向け並列言語の提案

小山浩生†　緑川博子†

†成蹊大学　工学研究科

筆者らは，メタプロセスモデルというデータローカリティに着目した階層型共有メモリプログラミングモデルを提案し，これに基づくポータブルな API として並列 C 言語，MpC を開発してきた．MpC コンパイラは，MpC プログラムをクラスタ向けには SDSM プログラムへ，共有メモリマシンに対しては pthread へ変換して並列実行を行えるようにしている．すでに，MpC プログラムがコモディティクラスタ（図 1)や共有メモリマシン(図 2)上で，UPC や OpenMP プログラムと比較し性能上の優位性があることを示した[1].

最近ではコア数の多いマルチコアクラスタ（図 3)が広く普及し始めている．そこで，マルチコアの特性を生かし MpC プログラムの性能をさらに引き出すために，図４のように，クラスタノードにまたがるプロセスとノード内にあるスレッドを組み合わせ，２レベルの並列処理機構を取り入れた共有メモリ型並列プログラミングモデルを新たに提案する．今回の実装では,プロセスローカルデータにアクセスする for 文を対象にしたスレッド並列構文 thread_for 文を MpC 文法に追加し， この区間に限ってスレッド並列を行う．あらかじめ MpC コンパイラにより，fork-join 方式でスレッド生成し，for 文におけるイテレーション区間を実行時に指定したスレッド数で等分して並列処理するコードに書き換えておく．（図 5) 現在のところ，thread_for の指定はユーザの責任で行い，for 文内のデータ，フローの依存性はコンパイラは関知しない．MpC プログラムでは thread_for 文以外のスコープで従来通りプロセス共有データ（shared）にアクセスすることが可能であるが，現段階では個々のスレッド（マスタースレッド以外）が直接 shared データにアクセスすることは，禁止している．

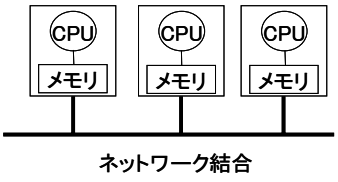

図１ 分散メモリ

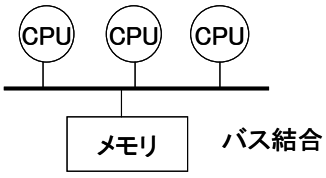

図２ 共有メモリ

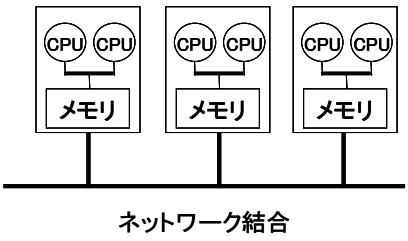

図３ マルチコアクラスタ

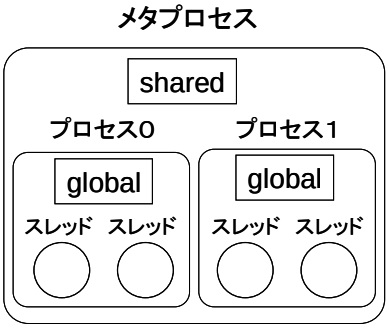

図４ プログラミングモデル

```
#define MAX 1000
int dummy[MAX];

int main()
{
   int i, var = 100,start,end;
   ・・・
   start = MAX/NPROCS*MYPID;
   end = MAX/NPROCS*(MYPID+1);
   thread_for（i = start; i < end; i++）{
      dummy[i] = i;
      dummy[i] += var;
   }
   ・・・
}
```

mpc コンパイラでの変換

```
int __param[32][2]
int dummy[1000];
void sub(void *__arg)
{
   int MYTID = (int )__arg[0];
   int i = *(int *)__arg[1];
   int var = *(int *)__arg[2];
   int start = __param[MYTID][0];
   int end = __prama[MYTID][1];
   int sub_start = start+MYTID*(end-start)/NTHREADS;
   int sub_end = start+(MYTID+1)*(end-start)/NTHREADS;
   for（i = sub_start; i < sub_end; i++）{
      dummy[i] = i;
      dummy[i] += var;
   }
}
int main()
{
   int i, var = 100,start,end;
   void **__arg[32];
   int __mpc_i;
   ・・・
   start = 1000/NPROCS*MYPID;
   end = 1000/NPROCS*(MYPID+1);
   for（__mpc_i = 0; __mpc_i < NTHREAD; __mpc_i++）{
      __param[__mpc_i][0] = start;
      __param[__mpc_i][1] = end;
      __arg[__mpc_i] = (void **)malloc(sizeof(void *) * 3);
      __arg[0] = (void *)__mpc_i;
      __arg[1] = &i;
      __arg[2] = &var;
      pthread_create(・・・, sub, arg);
   }
   /* スレッドを join する */
   ・・・
}
```

図５ プログラムの変換例

参考文献

[1] 緑川, 飯塚 :"メタプロセスモデルに基づくポータブルな並列プログラミングインターフェース MpC",情処論文誌：ACS, Vol.46 No.SIG4(ACS9), pp.69-85,（2005)