実験について

使用するファイル

1. Makefile コンパイル
2. MT.h メルセンヌツイスター
3. Rand.c タスクセットの生成（ファイル操作がスパゲティ）
4. LLF.c LLFスケジューラ
5. LMCF.c LMCFスケジューラ
6. LMCLF.c LMCLFスケジューラ
7. AcII\_LMCF.c AcII-LMCFスケジューラ
8. AcII\_LMCLF.c AcII-LMCLFスケジューラ
9. LLF\_LMCLF.c LLF LMCLF-combinedスケジューラ
10. LLF\_AcII\_LMCLF.c LLF AcII-LMCLF-combinedスケジューラ
11. Result.c 結果を出力するプログラム
12. cycle.pl 3.~8.を試行回数だけ実行（途中経過を出力）

* distribution.c タスクセットの利用率の分布調査用（実験には不要）
* Distribution.pl タスクセットの利用率の分布調査用（実験には不要）

実行するとできるファイル

（タスクセットのパラメータ）

1. rand\_period\_tasks.txt 各タスクの周期（行がタスク番号）
2. rand\_utilization\_tasks.txt 各タスクの利用率（行がタスク番号）
3. rand\_wcet\_tasks.txt 各タスクの最悪実行時間（行がタスク番号）
4. rand\_memory\_task\*.txt タスク\*の消費メモリ増分の時系列変化

実行するとできるファイル

（結果，最悪メモリ消費量とデッドラインミスとスケジュールのスイッチ）

1. datafile\_LLF.txt WCMC，DM
2. datafile\_LMCF.txt WCMC，DM
3. datafile\_LMCLF.txt WCMC，DM
4. datafile\_AcII\_LMCF.txt WCMC，DM
5. datafile\_ AcII\_LMCLF.txt WCMC，DM
6. datafile\_ LLF\_LMCLF.txt WCMC，DM，SW1
7. datafile\_ LLF\_AcII\_LMCLF.txt WCMC，DM，SW2
8. datafile\_ LLF\_AcII\_LMCLF.txt 個々のタスクの利用率の平均，利用率の最大
9. log.txt 結果のログ

※3.と5.~8.は末尾にαの値がついている．

実験方法（連続で複数回行う時）

動作環境：cygwin，WSL上（Linuxもok）で動く

1. 3.~10.のプロセッサ数を揃える（任意の数）
2. makeコマンドでコンパイル（make cleanで初期化）
3. cycle.plを実行

※デフォルトでは試行回数100になっているので，任意で指定

※実行する前に，Task\_Set\_\*というディレクトリがあるときは全て削除しておくこと（前回の実験結果が入っているとまずい）

※ちなみに，実験に時間がかかるため，実験終了後に，自動的にメールが送信されるようになっている． SMTPサーバのアドレスが変わっていなければ，自分の研究室のアドレスを入力するだけでよい．

実験方法（挙動を確認したい場合）

1. 上記の(ア)～(イ)までを行う．
2. ./randを実行する．（タスクセットを作る）

タスクセットがない状態で(ウ)のいずれかをすると，コアダンプする

1. その後，

./LLFでLLF

./LMCFでLMCF

./LMCLFでLMCLF

./AcII\_LMCFでAcII-LMCF

./AcII\_LMCLFでAcII-LMCLF

./LLF\_LMCLFでLLF LMCLF-combined

./LLF\_AcII\_LMCLFでLLF AcII-LMCLF-combined

の結果を見れる．

コードの内容についてはコメントを見ること

コード中にマクロされているが使用していない変数があるかもしれない（ｺﾞﾒﾝﾅｻｲ）

実験画面

スクリーンショット が含まれている画像

自動的に生成された説明

こんな感じで，

ディレクトリ

がいっぱいできる

この中に実験した

タスクセットのデータが入っている

実行すると，フリーズすることがある．

しかし，裏で重たいプロセスが動いているだけので，気長く待てば動き出す．

実行したらとりあえず放置．

また，重たいプロセスが動くため，CPUに大きな負荷がかかり，ほかの作業ができなくなる可能性があるので要注意．

実験条件

|  |  |
| --- | --- |
|  | 分布 |
| 消費メモリ増分MCI | 一様分布[-10000,10000] |
| 個々のタスクの利用率U | 指数分布(0.1,0.3,0.5,0.7,0.9) |
| 周期＝相対デッドライン | 一様分布[100,1000] |

最悪実行時間Cについては，利用率=最悪実行時間/周期であることから，利用率\*周期の値を用いた．ただし，利用率が1を超える場合はそのタスクを破棄する．また，利用率は実数なので，最悪実行時間が整数になるように，積を取った後，丸める．

実験の例

Task1 T1=6, C1=2, U1=1/3

Task2 T2=12, C2=6, U2=1/2

相対デッドラインが最大のタスクが終了するまでに，そのタスクの処理がほかのタスクにどれだけ邪魔されるか．

Task 2が最大の相対デッドラインをもつ．Task 2が終了するまでにTask 1が何度リリースされるかを考える．

つまり，周期12でTask1は0と6でリリースされる．

プログラム上ではリリースごとにタスクを分けて考えている．

したがって，実験の例のタスクセットではタスク数が2であるが，

0でリリースされるTask1とTask2と6でリリースされるTask1というように，スレッドの数は3となる．

LMCLFに用いられる優先度関数のは，プログラムの変数alphaでマクロされており，任意で変更可能．

変更したもので比較実験をするため，no.1からno.3まである．