# job自動実行システムjobautoのアイデア

- ローカルで管理する。計算サーバーを用意する(qsubシステムを想定）。
  できるだけ単純なデザインにしたい。

- qsubファイルは大量に作ることになる。たとえばQSGWのイテレーションごとに作るとすると、物質数xイテレーション数、のqsubファイルができる。qsub.{id}

- リモートディレクトリ REMOTE `yyy/foobar/`からローカルディレクトリ LOCAL `xxx/foobar/` に(必要なファイルだけ）rsyncしてlocalで制御する。qsub.0を初期ファイルとしてこれを逐次投入する。
  finishedはqsub.0により生成するファイルで判定する。qstatは用いない。

## リモートへの要求

- ecaljをインストールしておきパスを通しておく.
- sshでqsubできること。sshでrsyncできること。

## ローカルとリモート初期化

1. 事前準備のコードにより、`LOCAL/foobar`以下に初期ディレクトリ、初期ファイルを置いておく。

- いまの事前準備のコードinitposcarはPOSCARをPOSCARALLから分配してLDA計算させる場合。
- 初期化のスクリプトは仕事による。たとえばPOSCARを配置しておいて、LDAとQSGWを実行してバンドプロットする仕事など。

2. 事前準備においては各ディレクトリに以下の２つのファイルは必ず置く。

- qsubスクリプト： `qsub.{id}`
- qsub依存性ファイル: `qsub.dependency.{id}`
  qsub.{id}はディレクトリに複数個あっても良い。
  idは0,1,2,...でいい（なんでもいい）.
- qsub.{id}は終了時には`qsub.finished.{id}`というファイルを作って終了するようにしておく。
- qsub.dependency.{id}の中身には,qsub.{id}をスタートするのに必要なファイルを１行に１ファイル名で羅列しておく。
  (単純な例: qsub.dependency.{id}の中身をqsub.finished.{id-1}としておく. qsub.dependency.0は空にしておく）

## ジョブ自動管理のスクリプト jobmon.py

以下をローカルで--interval=time秒ごとに行う)。ジョブ管理スクリプトjobmon.pyをnohupで流す（将来的にはデーモンにしてもよいが、nohupでやれるならそれでいい）。すなわち、jobtestSGAにあるように

```
>nohup jobmon.py --ldir=LOCAL/foobar --rdir=REMOTE/foobar --binpath=binpath
...
```

として起動する。
ここでbinpathにはbinaryの入るディレクトリ名、pythonpathには用いるpythonへのpathが入るとする

- LOCAL/foobar ローカルマシンの計算パス。
  最低限の結果をシンクロしてリモートからコピーしてくる。quelistはログファイルであり、内部状態を監視するファイルでもある。
- LOCAL/foobar/以下のディレクトリで計算は行う。
  REMOTE/foobar リモートマシンの計算パス。
- --user=ユーザー名
- --remode= リモートマシン名
- --maxqsub=最大qsub数
- --maxcore=最大mpi数
  `jobmon.py --help`参照

## quelistのデザイン

各行に
`que名 [started@date] [finished@date]`
となる。再起動するときには`started`を`spre`に書き換えておく。`started@date`はqsubされたときに追記される。
`finished@date`はfinishedファイルが見つかったときに追記される。

## jobmonのアルゴリズム

1. 初期シンク

- LOCAL/foobar/lfoobar/initを
  LOCAL/foobar/lfoobar/{date} にコピーしたのち、
  REMOTE/foobar/lfoobar/date へrsyncする。
  {date}へは投入時点での日時を代入。
  コピーした後,qsub.{id}という形のqsubファイルについては,そのファイル内の文字列__binpath__をbinpathに,__pythonpath__をpythonpathに,__maxcore__をmaxcoreで置き換える。--initonlyで起動すると初期シンクのみで終了するとする。
- quelistを初期化（空リスト）。既存の場合は読み込む。

2. 以下は --intervalごとに繰り返す。

- rsyncをローカルで起動し、リモートのlfoobar/{date}以下のqsub.finished*について更新がある場合にローカルにコピーする。
- ローカルでqsub.dependency.*{id}を見て実行可能なものをLOCAL/foobar/{date}/quelistの末尾に追加する。
- finishedファイルが見つかれば、quelistに`finished@date`を追記
- 現在submitされてる数はquelistで「startがありfinishedがない」条件で探せる。

- qsub数が指定した最大数より小さい時,quelistを上から見ていってstartedしてないものをリモートでqsubする。quelistの同一行にstartedと追記.

- quelistをファイルに書き込む
- quelistのすべてがfinishedになれば終了

## ジョブ操作

- モニタする：foobar/date/quelistを見る.started finishedが見れる。またconsole出力でもカウントしている。
- 再起動：`--datedir=date` を加えて起動する。dateディレクトリが既存の場合、初期シンクはスキップする。前回終了時のquelistが使われる。きちんと初期化したい場合、qsub.0の冒頭にrm mix* rst*などと書いておく。
- 途中で終わった場合に --initonlyでrstなどをとってこれる。

## 注意点

- エラー処理はしていない。エラーシグナルはqsub.0で出すこと。たとえばfailファイルを出すようにすれば良い。ただしfinishedは残したほうが良いだろう。（終了と、エラーか正常か？は別の概念）。
-