TECHILA

With MATLAB

Contents

- **Short introduction on the main interfaces**
  - Peach
  - GridFor
- **Preparation**
  - Adding the Techila functions to the MATLAB search path
- **Creating a test Project**
- **Peach feature demonstrations based on requests**
  - Streaming & Callback
  - Job Input Files
  - Precompiled Binaries

Main interfaces - Peach

- **Simple interface for distributing computations**
- **At its simplest form, the peach function call typically defines**
  - The name of the function that will be deployed and executed on Workers
  - Input arguments for the executable function
  - List of data files that should be transferred to the Workers (optional)
  - Number of Jobs in the Project

```
function run_test()
var1=5;
result = peach('workerFunction',...   % Function that will be deployed and executed
               {2,var1,'<param>'},... % Input arguments for the function
               {'datafile.txt'},...   % Files that will be transferred to Workers
               1:10)                  % Number of Jobs in the Project
end
```

Main interfaces - Peach

- **Optional peach features can be used by adding additional parameters**
  - Streaming
  - Callback functions
  - …

```
function run_test_grid()
var1=5;
result = peach('workerFunction',...        % Function that will be deployed and executed
               {2,var1,'<param>'},...      % Input arguments for the function
               {'datafile.txt'},....       % Files that will be transferred to Workers
               1:10,...                    % Number of Jobs in the Project
               'StreamResults', 'true',... % Enable result streaming
               'CallbackMethod', @cbFunc)  % Call 'cbFunc' for each result file

end

function result=cbFunc(file)    % Callback function definition
result=load(char(file),'-mat') % Load the content of the result file into a struct
end
```

Main interfaces - GridFor

- **Provides a simple way to distribute for-loop structures**
  - Only use on computationally intensive loops
  - Iterations must be independent
  - GridFor blocks marked with 'gridfor' and 'gridend' keywords

```
function for_loop()
result=zeros(1,10);
for k=1:10
    result(k)=k*k;
end
end
```

Conversion possible

```
function gridfor_loop()
result=zeros(1,10);
gridfor k=1:10
    result(k)=k*k;
gridend
end
```

```
function for_loop()
result=0;
for k=1:10
    result=result+result*k;
end
end
```

Conversion NOT possible

```
For loop has recursive
dependancies. GridFor
cannot be used.
```

Main interfaces - GridFor

- **Default behaviour**
  - All defined Workspace variables will be transferred to Workers
  - All changed Workspace variables will be returned from Workers
  - Number of iterations performed in one Job set based on an estimate
    - Prevents exceedingly short Jobs
  - Return values joined by replacing old values
    - Control parameters can be used to define different methods for managing return values
- **GridFor control parameters used for fine tuning the behaviour**
  - `%gf:controlparameter=<value>`
- **Also accepts Peach parameters**
  - `%gf:peach <peachparam>=<peachvalue>`

Main interfaces - GridFor

- **Example**
  - Define that two iterations should be performed in each Job
  - Compile the code on Workers using Remote Compilation
  - Sum the variables 'var' returned from Workers
  - Transfer the file named 'input_1.mat' to all Workers

```
function gridfor_loop()
result=zeros(1,10);
var=0;
gridfor k=1:10
    %gf:stepsperworker=2
    %gf:datafile={{'input_1.mat'}}
    %gf:peach RemoteCompile='true'
    %gf:sum=var
    result(k)=k*k;
    load input_1.mat
    var=var+k;
gridend
end
```

Peach

Original Code

```
...
for x=1:length(S0)
  for y=1:length(sigma0)
     price(y,x) = asian_montecarlo(S0(x),sigma0(y)^2,M,nn,r,N,rho,kappa,psi,E,T);
   end
end
...
function [price] = asian_montecarlo(S0,v0,M,nn,r,N,rho,kappa,psi,E,T)
...
```

Peach Control Code

```
...
price = peach('asian_montecarlo', {S0, sigma0.^2, M, nn, r, N, rho, kappa, psi, E, T,
'<param>'}, {}, 1:length(S0)*length(sigma0));
price = cell2mat(reshape(price,length(sigma0),length(S0)));
...
```

Peach Worker Code

```
function [price] = asian_montecarlo (Sx,vx,M,nn,r,N,rho,kappa,psi,E,T,jobidx)
[j, i] = ind2sub([length(vx), length(Sx)], jobidx);
S0 = Sx(i);
v0 = vx(j);
...
```

GridFor

Original Code

```
...
for x=1:length(S0)
  for y=1:length(sigma0)
     price(y,x) = asian_montecarlo(S0(x),sigma0(y)^2,M,nn,r,N,rho,kappa,psi,E,T);
   end
end
...
function [price] = asian_montecarlo(S0,v0,M,nn,r,N,rho,kappa,psi,E,T)
...
```

GridFor Code

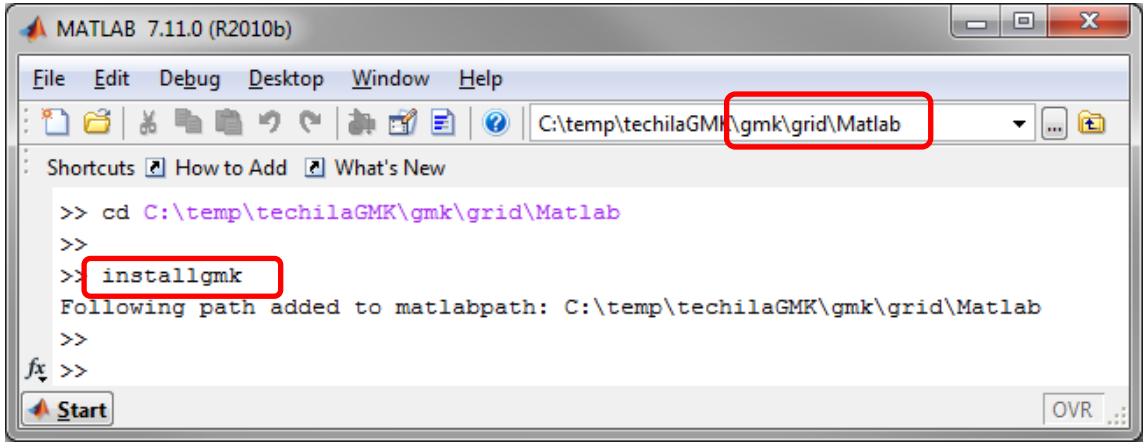
```
...
gridfor x=1:length(S0)
   gridfor y=1:length(sigma0)
     price(y,x) = asian_montecarlo(S0(x),sigma0(y)^2,M,nn,r,N,rho,kappa,psi,E,T);
   gridend
gridend
...
function [price] = asian_montecarlo(S0,v0,M,nn,r,N,rho,kappa,psi,E,T)
...
```

Preparation

- **Adding the Techila functions to MATLAB search path**
  1. Launch MATLAB
  2. Change your current working directory to:

     <full path>\gmk\grid\Matlab
  3. Execute command:

     `installgmk`



- **Access help with commands:**
  - doc gridfor
  - doc peach
  - doc remotecompile

Creating a Project for testing purposes

1. **Change your current working directory in MATLAB to:**

   <full path>\gmk\examples\Matlab\Tutorial\1_gridification

   Note! If you're using Windows or Mac, run the Remote Compilation example at:

   <full path>\gmk\examples\Matlab\Features\remote_compiling

   ```
   result = run_remote_pi(10, 1000000)
   ```

2. **Create the test Project with command:**

   ```
   result=run_gridification(5)
   ```

3. **When prompted, enter your password**

4. **A status bar will be displayed containing information of the Project**

Requests on what features to demonstrate

- **Techila Grid Management Kit contains examples of:**
  - Snapshots
  - Streaming & Callback
  - Job Input Files
  - Precompiled binaries
  - Iterative Projects
  - And more..

WWW.TECHILA.FI