Changes in the LPJmL Code

Werner von Bloh

Potsdam Institute for Climate Impact Research

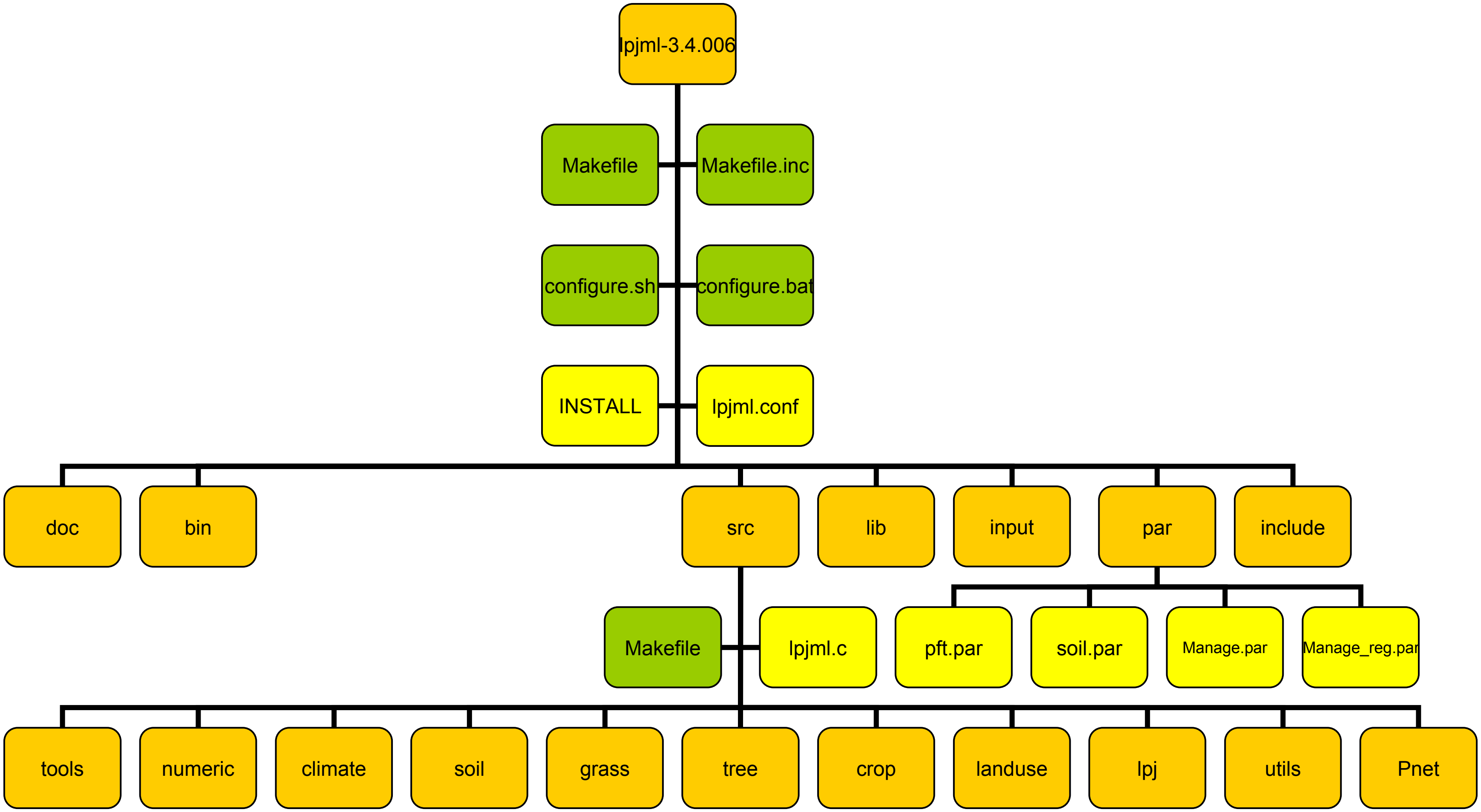
Major changes

- MPI parallelism included necessary for coupling with Climber-2
- Advantage:
  - Automatic distribution of data
  - Easy handling, only one restart file and output file for each variable created.
- Disadvantage:
  - Load imbalance (improved by random shuffling of data, not possible in case of river routing)
- `configure.sh` script is modified to check for parallel environment (AIX, Linux, Mac OS X)

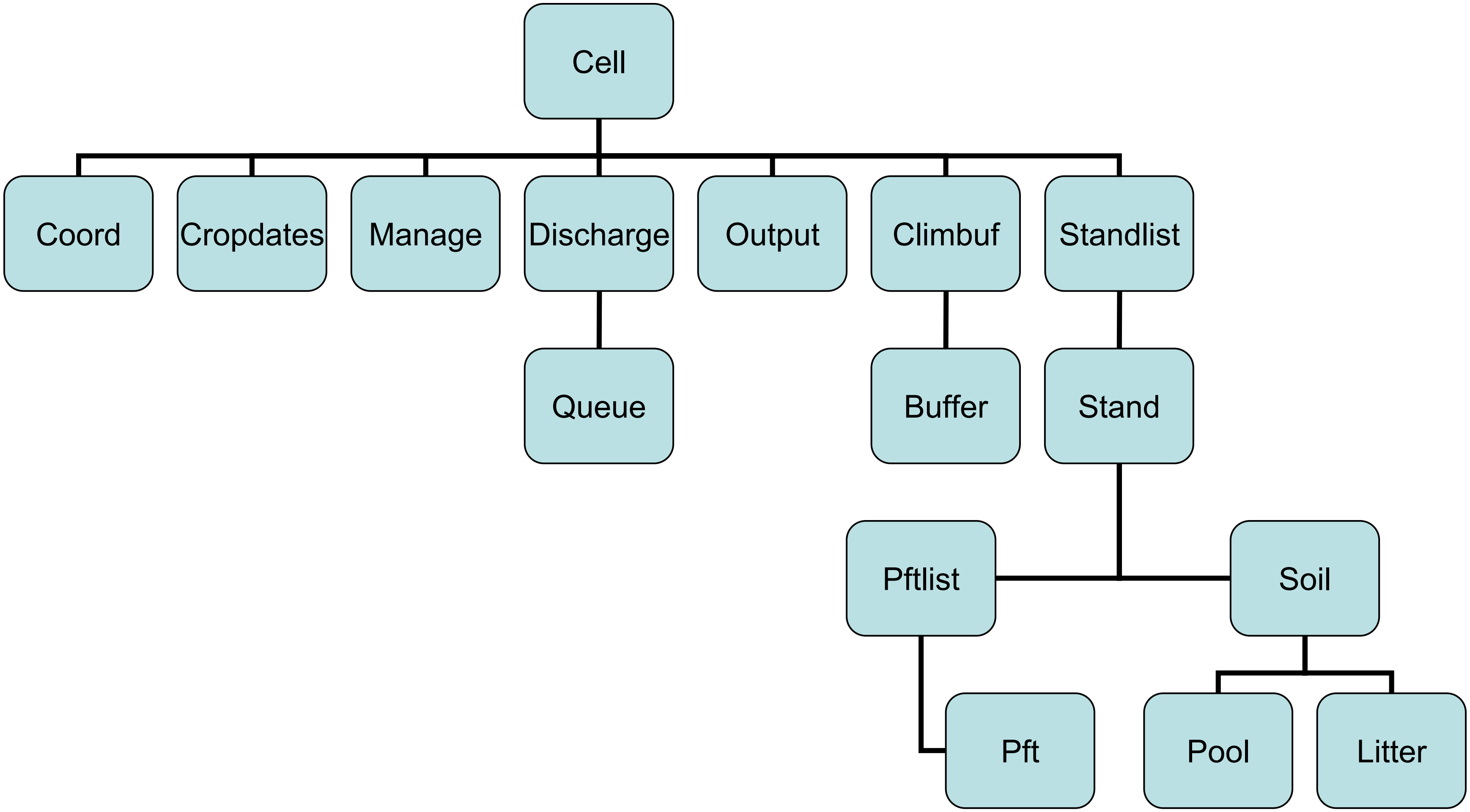
Minor changes

- Two versions of `iterateyear`:
  Without river routing loop over one year is done for each grid cell separately. Improves data locality
- Reproduction and turnover function merged
- Datatype `Harvest` and `Wateruse` included
- Try to minimize function parameters
- Datatype `Config` contains now `Pftpar` and `Soilpar`
- Often used expressions put in `Pftpar`
- Output is buffered in memory `(-DBUFFER`)
- Climate data is stored in memory for spinup phase
- More variables written in restart file including crops

Directory structure



Datatype Cell



Datatype Pft

```
typedef struct Pft
{
  const struct  Pftpar
  {
    int id;      /* unique PFT identifier */
    int type;    /* type --> whether CROP or TREE or GRASS*/
    char *name;  /* Pft name */
    Real gmin;   /* canopy conductance component (4)  */
    /* ... */
    void *data;  /* pointer for PFT specific extensions */

    /* list of pointers for PFT specific functions */

    void (*newpft)(struct Pft *);
    /* .... */
  } *par;  /* Pft parameters */
  Real fpc;   /* foliar projective cover (FPC) under full leaf
                 cover as fraction of modelled area */
  /* ... */
  void *data; /* pointer for PFT specific extensions */
} Pft;
```

Virtual PFT functions

```
void (*newpft)(struct Pft *);
void (*init)(struct Pft *);
Real (*turnover)(Litter *,struct Pft *);
Real (*npp)(struct Pft*,Real,Real,Real,Bool *,Real);
Real (*fpar) (const struct Pft*);
Bool (*leaf_phenology)(struct Pft *,Real,Real,const Real[],int,
                       Real,Bool,Landusetype);
Bool (*fwrite)(FILE *,const struct Pft *);
void (*fread)(FILE *,struct Pft *,Bool);
void (*fprint)(FILE *,const struct Pft *);
Harvest (*litter_update)(Litter *,struct Pft *,Real,Bool);
Bool (*allocation)(Litter *,struct Pft *,Real *);
Real (*establishment)(struct Pft *,Real,Real,int);
Bool (*mortality)(Litter *,struct Pft *,Real,Real);
Real (*fire)(struct Pft *,Real *);
Real (*lai)(const struct Pft *);
void (*adjust)(Litter *,struct Pft *,Real);
void (*free)(struct Pft *);
void (*light)(Litter *,struct Pft *,Real);
Real (*vegc_sum)(const struct Pft *);
void (*mix_veg)(struct Pft *,Real);
```

Compiling the code

- Run
  - `./configure.sh` (on AIX/Linux/Mac OS X)
  - `configure.bat` (on Windows)
- Software needed for Windows (available from http://www.microsoft.com):
  - Microsoft C++ Compiler
- Run `make` to create executable. Executable will be put in the bin directory.

Adding new PFT classes

- New PFT design template is built by invoking the newpft.sh script:
  `% bin/newpft.sh $pft`
- Directory `src/$pft` is created including Makefile and function templates for all virtual PFT functions, e.g. `fire_$pft.c`
- Header file `$pft.h` is created in include
- Initialization function `fscanpft_$pft.c` is defined

Utilities

Utility programs can be created by `make utils`

Filename	Description
`catlpj`	concatenates LPJ restart files
`printlpj`	prints content of restart file to stdout
`shuffle`	shuffles input data randomly to improve efficiency of parallel code. River routing must be disabled.
`cru2clm`	converts CRU data into file format suitable for LPJ
`txt2clm`	Converts CRU data files into LPJ climate data files CRU data files have to be in the format specified in http://www.cru.uea.ac.uk/~timm/grid/CRU_TS_2_1.html

Running LPJmL

- Invoke
  ```
  % bin/lpjml lpjml.conf
  ```
  where lpjml.conf is the (default) configuration file
- `lpjml.conf` and all other input files are piped thru the cpp preprocessor
- If lpjml is executed outside the root directory of LPJ the environment variable LPJROOT has to be set:
  ```
  % export LPJROOT=<lpj root dir>
  ```
  Then path to include files is added.
- Macros for preprocessing the configuration file can be defined on the command line:
  ```
  % bin/lpjml -DFROM_RESTART lpjml.conf
  ```

LPJ configuration file

```
#include "include/conf.h" /* include constant definitions */
"LPJmL run with river routing and lakes" /* Simulation description */
LPJML             /* Simulation type with managed landuse */
#include "par/pft.par" /* PFT Parameter file */
#include "par/soil.par"  /* Soil Parameter file */
#ifdef WITH_LANDUSE
LANDUSE  /* landuse changes enabled (LANDUSE/NEW_LANDUSE) */
#include "par/manage.par" /* Management Parameter file */
#include "par/manage_reg.par" /* Management Parameter file */
#else
#NO_LANDUSE
#endif
#include "cru.conf"
#ifdef ISRANDOM
RANDOM_SEED
#endif
2                 /* number of output files */
/*
ID           filename
------------ ---------------------------- */
GRID         output/grid.bin
VEGC         output/vegc.bin              /*
------------ ----------------------------- */
```

LPJ configuration file cont‘d

```
FIRE  /* fire disturbance enabled */
0     /* first grid cell */
59198 /* last grid cell */
#ifndef FROM_RESTART
500  /* spinup years */
30   /* cycle length during spinup (yr) */
1901 /* first year of simulation */
1901 /* last year of simulation */
NO_RESTART /* do not start from restart file */
RESTART    /* create restart file */
restart/restart_1900.lpj  /* filename of restart file */
#else
0    /* no spinup years */
1901 /* first year of simulation */
2003 /* last year of simulation */
RESTART /* start from restart file */
restart/restart_1900.lpj  /* filename of restart file */
NO_RESTART /* do not create restart file */
#endif
```

Input file configuration

```
#include "include/conf.h" /* include constant definitions */
0.5  0.5 /* resolution in degrees */
input/grid.bin
input/soil.bin               /* soilcode data */
#ifdef WITH_LANDUSE
input/cow_2006.bin           /* country and region codes */
input/luc_gmia3.clm          /* Landuee data */
#endif
#ifdef RIVER_ROUTING
DRAINAGE
input/lakes.bin              /* lake fractions */
input/drainage.bin           /* river routing */
input/neighb_irrigation.bin
#else
NO_DRAINAGE
#endif
input/tmp.clm                /* temperature data */
input/pre.clm                /* precipitation data */
input/cld.clm                /* cloudiness data */
input/co2_2003.dat           /* CO2 data */
#ifdef ISRANDOM
RANDOM_PREC /* Random weather generator for precipitation enabled */
input/wet.clm                /* number of wet days */
#else
INTERPOLATE_PREC /* interpolate daily precipitation */
#endif
```

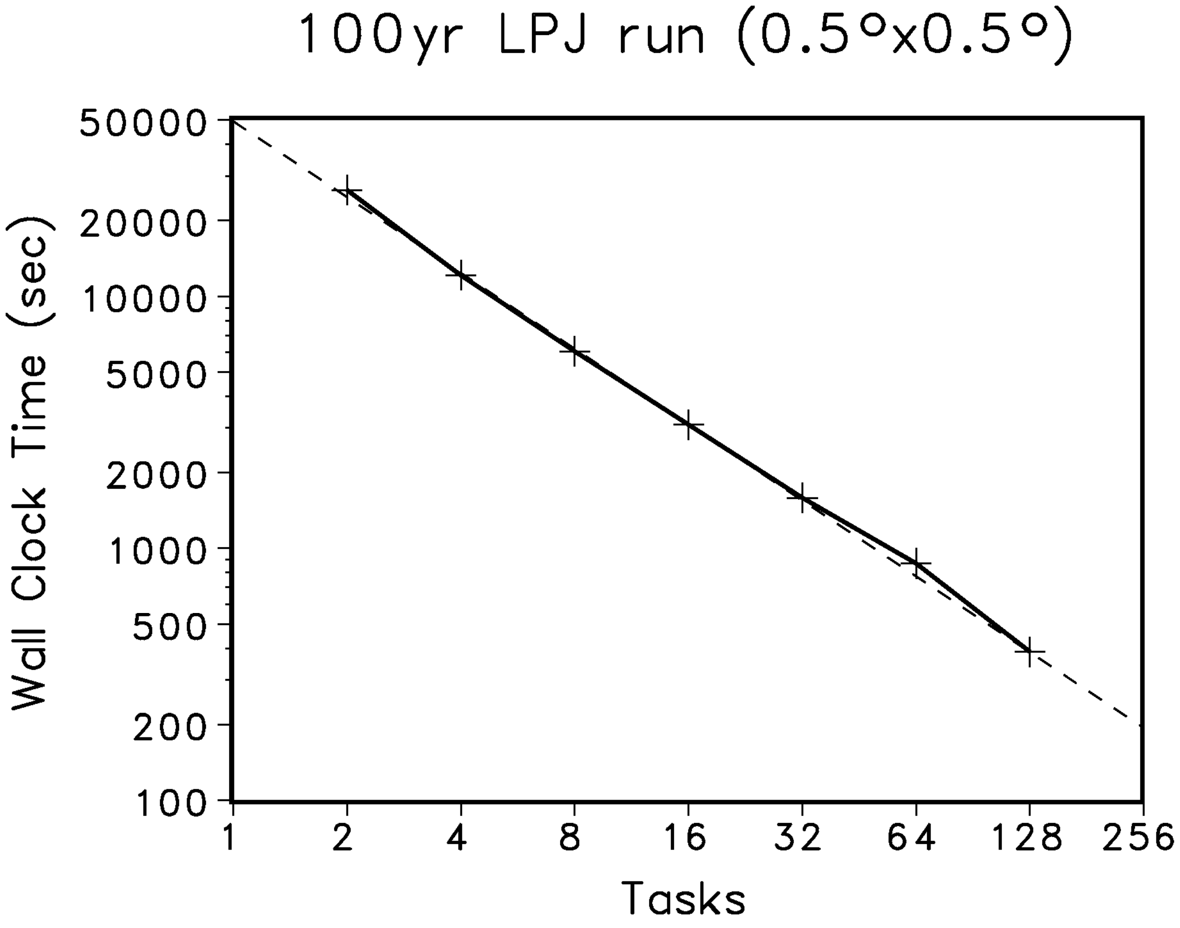
Sample output

```
% mpirun -np 64 bin/lpjml -DFROM_RESTART -DWITH_LANDUSE -DRIVER_ROUTING
**** ./lpjml C Version 3.4.006 (Aug 13 2008) Linux ****

Reading configuration from ‘lpjml.conf' with options '-DFROM_RESTART‘,’-DWITH_LAN
DUSE’,’-DRIVER_ROUTING’.
Simulation "LPJmL run with river routing and lakes" running on 64 tasks
Starting from restart file 'restart/restart_1900.lpj'.
Input files:
Variable  Filename
--------- ---------------------------------------------------------------------
temp      input/temp_shuffle.clm
prec      input/prec_shuffle.clm
cloud     input/cloud_shuffle.clm
countries input/country_shuffle.clm
landuse   input/landuse_shuffle.clm
co2       input/co2_2003.dat
--------- ---------------------------------------------------------------------
Number of output files: 2
Byte order in output files: little-endian
Variable      Filename
------------- ---------------------------------------------------------------
         grid output/grid.bin
         vegc output/vegc.bin
------------- ---------------------------------------------------------------
No spinup years.
First year:     1901
Last year:      2003
Simulation begins...

Year     NEP     fire    estab  harvest  total
------ ------- ------- ------- ------- -------
  1901  11.326   4.141   0.202   5.650   1.736
  1902  10.534   4.150   0.206   5.351   1.238
```

Parallel version

- Parallel version is enabled via `-DUSE_MPI` flag. The message-passing libary (MPI) has to be installed on the computer (checked by `configure.sh`)
- Communication in the river-routing version is performed with the help of the Pnet library.
- LoadLeveler job control files `(*.jcf`) are provided for the PIK Linux and AIX cluster.

Efficiency with river routing



Run time on a 3 GHz Xeon Intel cluster with Infiniband network.