# SPPARKS for microstructural science

Brian DeCost

11 July 2014

```
#include "string.h"
#include "spparks.h"
#include "memory.h"
#include "error.h"
#include "universe.h"
#include "input.h"
#in
#in
#in
#include "potential.h"
#include "random_mars.h"
#include "timer.h"
#include "output.h"

using namespace SPPARKS_NS;

/* ----------------------------------------------------------------------
   allocate fundamental classes (memory, error, universe, input)
   parse input switches
   initialize communicators, screen & logfile output
   input is allocated at end after MPI info is setup
------------------------------------------------------------------------- */

SPPARKS::SPPARKS(int narg, char **arg, MPI_Comm communicator)
{
  error = new Error(this);
```

# Stochastic Parallel PARticle Kinetic Simulator[1]





**SPPARKS is an efficient *parallel* Kinetic Monte Carlo framework**

- Mesoscale materials phenomena as stochastic processes
- Scalable to thousands of processors
- Modular C++ facilitates user-defined models/statistics
- Open Source (GPLv3)
- Related to the LAMMPS Molecular Dynamics framework

[1] http://spparks.sandia.gov/

# What applications is SPPARKS useful for?





- lattice-based models
- off-lattice models
- generic KMC applications

**In this session, we'll focus on the Potts model**

- Modeling grain growth, recrystallization, and related phenomena
- Annealing synthetic microstructures to achieve local equilibrium along triple lines
- Generation of simple synthetic microstructures

# Spatially decomposed parallelization

# The SPPARKS distribution





Download SPPARKS from
`http://www.sandia.gov/~sjplimp/download.html`

**Unpack the SPPARKS source archive**

```
mkdir -p ~/Software && cd ~/Software
tar xvzf ~/Downloads/spparks.tar.gz
cd spparks*/src
ls ~/Software/spparks*/
```

```
brian@eeepc:spparks-19Nov13 $ ls ~/Software/spparks-19Nov13
doc  examples  LICENSE  python  README  src  tools
```

# SPPARKS uses the Unix build system make





SPPARKS is distributed with several makefiles, found in src/MAKE/.

```
bdecost@unix6 > ls MAKE                              :~/Software/spparks-19Nov13/src
Makefile.blackrose  Makefile.mac        Makefile.serial
Makefile.cygwin     Makefile.mac_debug  Makefile.serial_backup
Makefile.g++        Makefile.mac_mpi    Makefile.serial_debug
Makefile.linux      Makefile.redsky     Makefile.xt5
```

You will likely have to edit Makefile.linux, Makefile.g++, or Makefile.mac_mpi to reflect your system to get SPPARKS to compile.[2]

[2] You'll likely need to supply the location of the MPI library files on your system.

# Building SPPARKS





Try to compile SPPARKS by running make with the appropriate build target for your system as an argument.

**Building SPPARKS using Makefile.mac_mpi**

```
make -j4 mac_mpi
```

The result is an executable named `spk_mac_mpi` in `src/`.

# Building serial SPPARKS





The JPEG library can also cause build problems which are difficult to resolve. Makefile.demo will build a serial version of SPPARKS without the JPEG library.

**First build the MPI stubs:**

```
cd STUBS
make
cd ..
```

Copy Makefile.demo into src/MAKE/

**Build the demo version of =SPPARKS=**

```
make demo
```

# "Installing" SPPARKS[3]





**To avoid copying the SPPARKS executable or typing the full path to it repeatedly, define a *shell alias*:**

```
alias spk='~/Software/spparks*/src/spk_demo'
```

Now typing `spk` into the shell prompt from any working directory runs SPPARKS. Try this now.

[3] To make this persistent, append this command to `~/.bashrc` (`~/.bash_profile` on a mac)

# Running SPPARKS with no input





## SPPARKS reads commands from stdin

```
brian@rsfern > spk
SPPARKS (19 Nov 2013)
```

Press Ctrl-C to exit.

# SPPARKS input scripts





SPPARKS is controlled by a script of commands which are executed line-by-line.

**Run the =basic= demo**

```
cd ~/Downloads/spparks-demo-master/demos/basic
spk < basic.spkin
```

# Anatomy of an input script: basic.spkin





```
# Minimal SPPARKS Potts script
seed                56789
app_style           potts 100
dimension           3
lattice             sc/26n 1.0
region              box block 0 32 0 32 0 32
create_box          box
create_sites        box
set                 site range 1 100
solve_style         tree
diag_style          energy
stats               10.0
run                 100.0
```

# Diagnostics

# Log file

# The dump command





Post-processing. `pizza.py` or custom scripts.

# Installing pizza.py[4]





Download and unpack the pizza.py archive from
`http://www.sandia.gov/~sjplimp/download.html`

**Unpack the SPPARKS source archive**

```
mkdir -p ~/Software && cd ~/Software
tar xvzf ~/Downloads/pizza.tar.gz
cd -
alias pizza='python -i ~/Software/pizza*/src/pizza.py'
```

[4] As with SPPARKS, append the `alias` command to `~/.bashrc` or `~/.bash_profile` to make this "installation" persistent.

# Conversion to vtk with pizza.py

# mpirun

# Processor partitions