GPU Computing with CUDA
Lab 2 - FD in global and texture memory

Christopher Cooper
Boston University

August, 2011
UTFSM, Valparaíso, Chile

Objectives

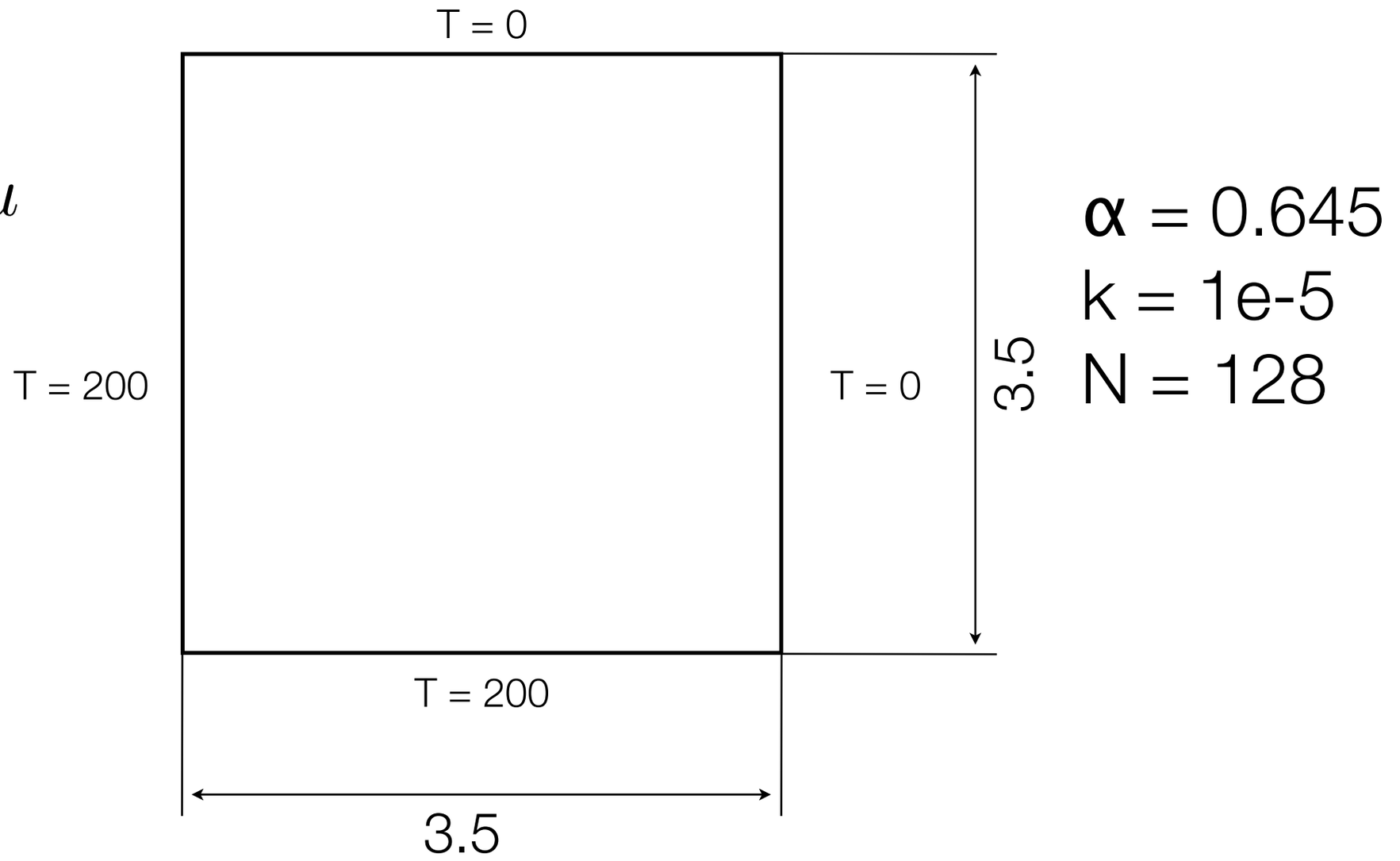
- Implement a simple finite difference problem in CUDA
- Learn how to use texture memory

```
wget http://www.bu.edu/pasi/files/2011/07/codes_lab21.zip
unzip codes_lab21.zip
```

Heat transfer with FD

- Heat diffusion on square 2D flat plate with Dirichlet boundary conditions and 0 initial condition

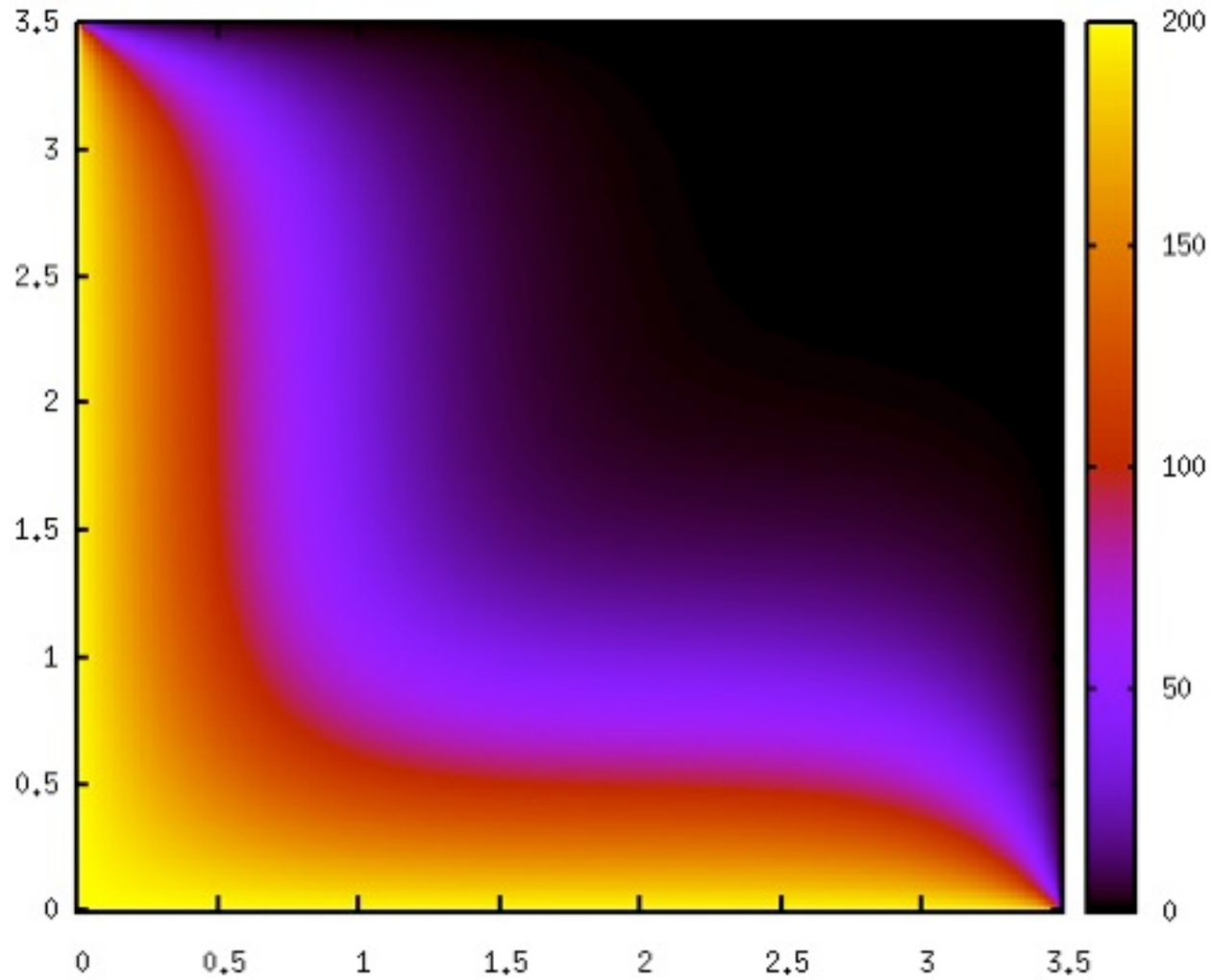
$$\frac{\partial u}{\partial t} = \alpha \nabla^2 u$$



$$u_{i,j}^{n+1} = u_{i,j}^{n} + \frac{\alpha k}{h^2}(u_{i,j+1}^{n} + u_{i,j-1}^{n} + u_{i+1,j}^{n} + u_{i-1,j}^{n} - 4u_{i,j}^{n})$$

Heat transfer with FD

- Each thread will do one mesh point
- Global memory data lives for whole execution!
- Experiment with other mesh sizes (N) that are not a multiple of block size



Heat transfer with FD

- Changes to global memory case: (start from pervious code!)

  - Initialize texture (beginning of code)

```
texture <float, 2> tex_u;
```

  - After allocation, bind texture to global memory

```
cudaChannelFormatDesc desc = cudaCreateChannelDesc<float>();
cudaBindTexture2D(NULL, tex_u, u_d, desc, N_max, N_max,
sizeof(float)*N_max);
```

- In the kernel, fetch values from texture memory

```
int x = threadIdx.x + blockIdx.x*BSZ;
int y = threadIdx.y + blockIdx.y*BSZ;
c = tex2D(tex_u_prev, x, y);
```

- Unbind texture at the end

```
cudaUnbindTexture(tex_u);
```

Heat transfer with FD

- Using texture memory:
  - Should give good results as it is designed for 2D data locality
- Textures map multiples of 32 or 64:
  - Need to pad or use cudaMallocPitch
- Texture memory is read only within a kernel
  - Need to write separate kernel to update values in texture