Simulation Ideen-Verbreitung

Projektvorstellung

Arne Struck, Jonathan Werner, Manuel Börries

Universität Hamburg, Fachschaft Informatik, Praktikum paralleles Programmieren

24. September 2014

Ziel

(Grobe) Simulation von Entwicklung konkurrierender Ideen in einer begrenzten Welt.

Population

Idee

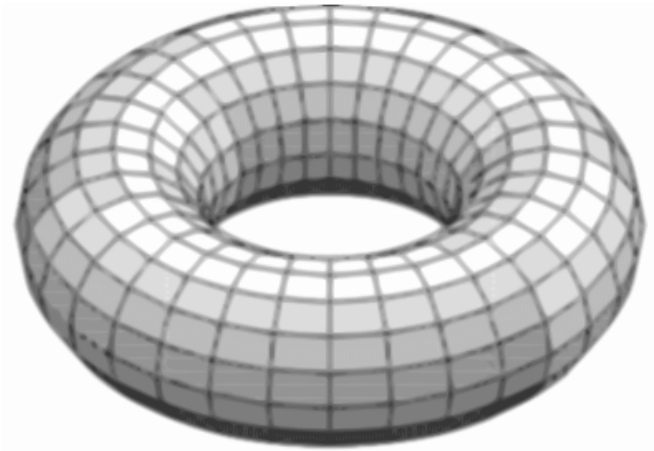
- Qualität
- Komplexität
- Weltanschauung

Mensch

- Idee
- Weltanschauung

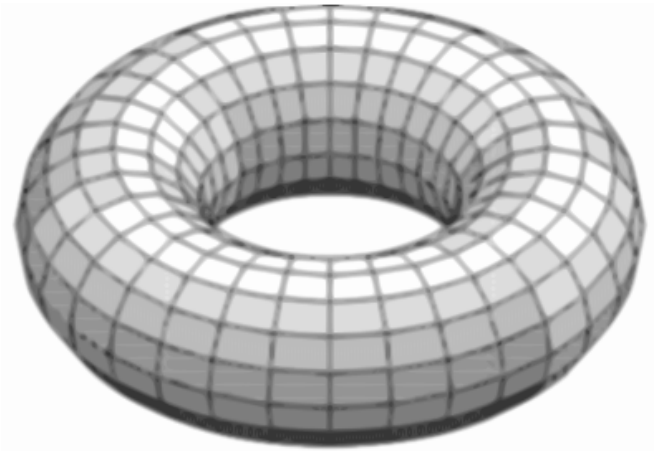
Welt & Bewegung

Die Welt

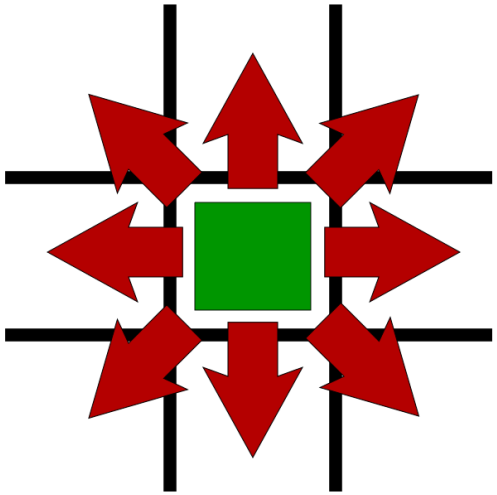


Welt & Bewegung

Die Welt

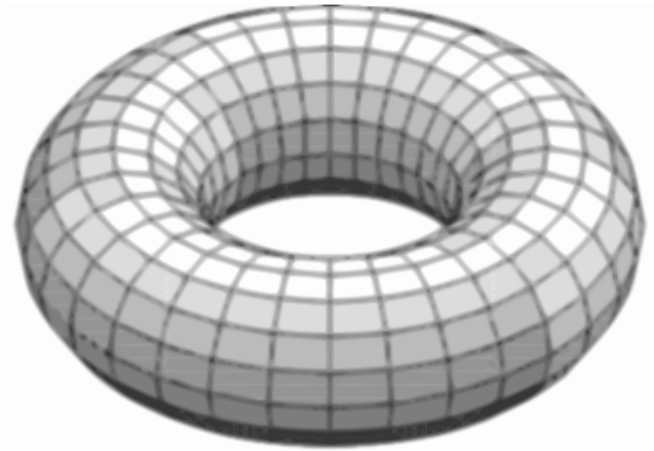


Bewegungsziele

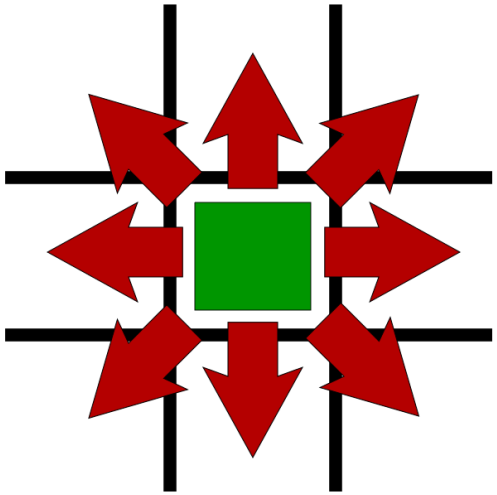


Welt & Bewegung

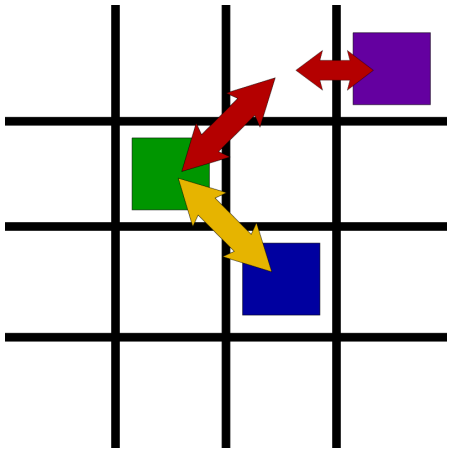
Die Welt



Bewegungsziele



Kommunikation



Kommunikation

3-Phasen:

1. Kompatibilitätscheck
2. Evaluation des Gewinners
3. Aufstellung der neuen Merkmale des Verlierers

Mutation

Qualität

- Wahl der Mutationsrichtung
- Mutation der Qualität
- Kaskadierend der Komplexität

Mutation

Qualität

- Wahl der Mutationsrichtung
- Mutation der Qualität
- Kaskadierend der Komplexität

Weltanschauung

- Wahl der Mutationsrichtung
- Mutation des Idee-Wertes
- Mutation des Mensch-Wertes
- Differenzcheck

Ablauf

- Initialisierung des Feldes
- Zufälliger Spawn der Menschen mit mehrheitlich geringen Qualitätswerten
- Beginn der Simulationsschleife für n Schritte
  - Mutationsevaluation
  - Kommunikationsversuch
  - Bewegung
- Ende der Schleife

Was ist eine Idee?

Eigenschaften einer Idee

- Qualität
- Komplexität
- "Weltanschauungswert"
- Komplexität abhängig von Qualität

Qualität

```
int chance = rand_int(1000, 0);
 if(chance < BEST_IDEA_CHANCE){
   i.a = rand_int(IDEA_MAX, 0);
 }
 else if(chance < MED_IDEA_CHANCE) {
   i.a = rand_int((int) (IDEA_MAX * 0.66), 0);
 }
 else {
   i.a = rand_int((int) (IDEA_MAX * 0.33), 0);
 }
```

Komplexität

```
int tempb = i.a + rand_int(2 *
    QUAL_CMPLXTY_DEP_RANGE + 1,
     -QUAL_CMPLXTY_DEP_RANGE);
  if (0 <= tempb) {
    if (tempb < IDEA_MAX) {
      i.b = tempb;
    }
    else i.b = IDEA_MAX - 1;
  }
  else i.b = 0;
```

Kommunikation & Gewinner Berechnung

1. Bewegen der Ideen (random)
2. Überprüfen ob Ideen miteinander in Wettkampf können

```
int convinceable = 1;
if(abs((i1.c - i2.h)) > MAX_CWV_VS_HWV || abs((
    i2.c - i1.h)) > MAX_CWV_VS_HWV) {
      convinceable = 0;
    } else if(complxdif > MAX_CMPLX_DIFF) {
      convinceable = 0;
```

3. Bestimmung der Winner-Idea
4. Überschreiben der Loser-Idea

Parallelisierungsschema

- Implementierung des Felds durch 2D-Array von structs (Ideas)
- Es werden zwei Felder erstellt:

```
malloc_idea_matrix(field)
malloc_idea_matrix(field_new)
```

- sich bewegte Ideen werden in neues Feld kopiert

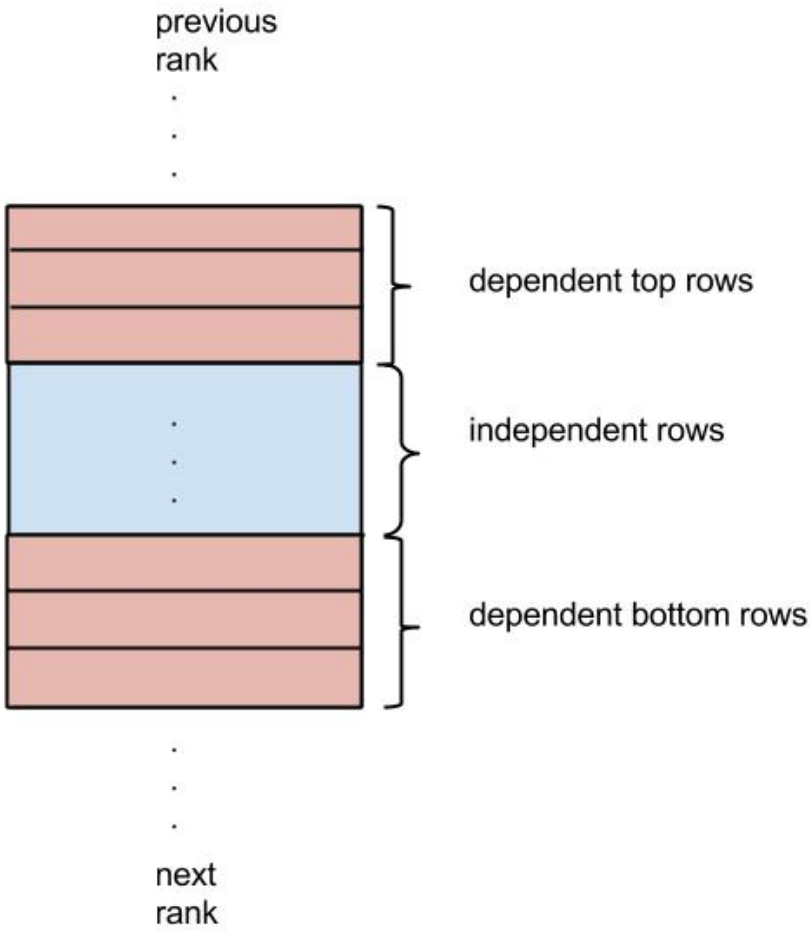
```
#define copy_field_into_field_new()
    for_every(i, num_rows, {
        for_every(j, num_cols, {
            field_new[i][j] = field[i][j];
        });
    });
```

Parallelisierungsschema

Parallelisierungsschema

- horizontales Kommunikationsmuster
- am oberen bzw. unteren Rand ghost row
- Ziehen unabhängig von anderen Prozessen möglich

Parallelisierungsschema



1. Bewegung der independent ideas
2. Bewegung der top dependent ideas, Kommunikation dieser
3. Bewegung der bottom dependent ideas, Kommunikation dieser

Parallelisierungsschema

```
#define send_ideas(ideas_arr, to, tag, req)
  MPI_Isend(ideas_arr, num_cols,
    mpi_idea_type, to, tag, MPI_COMM_WORLD, &req)

#define receive_ideas_into(ideas_arr, from, tag, req)
  MPI_Irecv(ideas_arr, num_cols,
    mpi_idea_type, from, tag, MPI_COMM_WORLD, &req)
```

Parallelisierungsschema

```
#define mpi_define_idea_type()
  int blocklengths[5] = {1,1,1,1,1};
  MPI_Datatype types[5] = {MPI_INT,MPI_INT,MPI_INT,
      MPI_INT, MPI_INT};
  MPI_Datatype mpi_idea_type;
  MPI_Aint offsets[5];
  offsets[0] = offsetof(Idea, a);
  offsets[1] = offsetof(Idea, b);
  offsets[2] = offsetof(Idea, c);
  offsets[3] = offsetof(Idea, h);
  offsets[4] = offsetof(Idea, empty);
    MPI_Type_create_struct(5, blocklengths, offsets
        , types, &mpi_idea_type);
    MPI_Type_commit(&mpi_idea_type);
```

Visualisierung

- lokale Visualisierung mithilfe von Pygame
- Pro Runde ein File
- im Nachhinein: Einlesen in Python
- Problem: Files werden zuerst auf dem Cluster gespeichert und muessen nach local kopiert werden

Optimierung

- Ersetzen von Send/Recv mit Isend/Recv (10 % Speedup)
-

Profiling

HIER EIN BILD VON MESSUNGEN

Tracing

idk, ob wir das machen wollen

Ablauf

kp, wie ich das einfügen machen. call auf player vllt noch, aber dann ist das systemabhängig

Ergebnisse

- Qualität nimmt über die Zeit zu
- Obwohl andere selten vollständig entfernt bilden 2-3 Ideen eine Majorität aus
- Qualität/Elaborationsgrad nimmt über die Zeit zu
- Es bleiben einige Menschen mit Ideen niedriger Qualität
- Selten: Durch Mutation entwickelt sich eine verdrängte Idee zur dominanten

Probleme

Probleme beim Debuggen

- Logik und Bewegung größtenteils unter Beteiligung von Zufallselementen
- oft nicht reproduzierbare Bugs

Real-Time Visualisierung

- große Datenmenge
- Uns war nicht klar wie/ob X-Forwarding mit dem Cluster funktioniert wird