epcc

- Feedback Guided Scheduling for Two-Dimensional Loops
- Oleh Olkhovskyy
- Supervisor: Mark Bull
- Edinburgh Parallel Computing Centre
- Thursday 30^{th} August 2001, 13:30

- Two-Dimensional loop means sequential outer loop and 2D inner parallel loop
- Loop scheduling is assignment of loop iterations to threads/processors to minimise overheads

- The four critical sections of loop scheduling algorithms are synchronisation, process management, communication and load imbalance
- Synchronisation occurs when processor must wait for some action by another processor, such as relinquishing a critical region
- Process management refers to time needed to calculate iterations boundaries of each processor

- Communication is interaction between processors
- Load imbalance occurs when some processors finish their calculations earlier than other processors

Loop scheduling

- Static and dynamic iteration distribution
- Static distribution means that each processor has explicitly defined iteration boundaries
- Dynamic distribution defers the assignment of iterations to processors until run-time

Loop scheduling algorithms

- 2 approaches of dealing with 2-dimensional loops are used here
  - 2-dimensional loop is treated as 1-dimensional loop
  - Using 2-dimensional loop scheduling algorithm

- Guided algorithm

  Each processor executes $\frac{R}{P}$ from common queue

- Affinity algorithm

  Each processor receives $\frac{n}{P}$ iterations

  Processor removes $\frac{1}{P}$ iterations of its local queue and executes them

  If processor's queue is empty, it finds processor with the most work left, and removes and executes $\frac{1}{P}$ part of batch

  On new turn, each processor receives as many iterations, as it calculated on previous turn

- Trapezoid algorithm
  - Each processor executes batch from common queue
  - Batch size is decreased linear, from f to l
  - Usually f assumed equal $\frac{n}{2*P}$ and l equal to 1
  - The number of batches is $N = \frac{2n}{f+l}$
  - Batch size is decreased by $\delta = \frac{f-l}{N-1}$

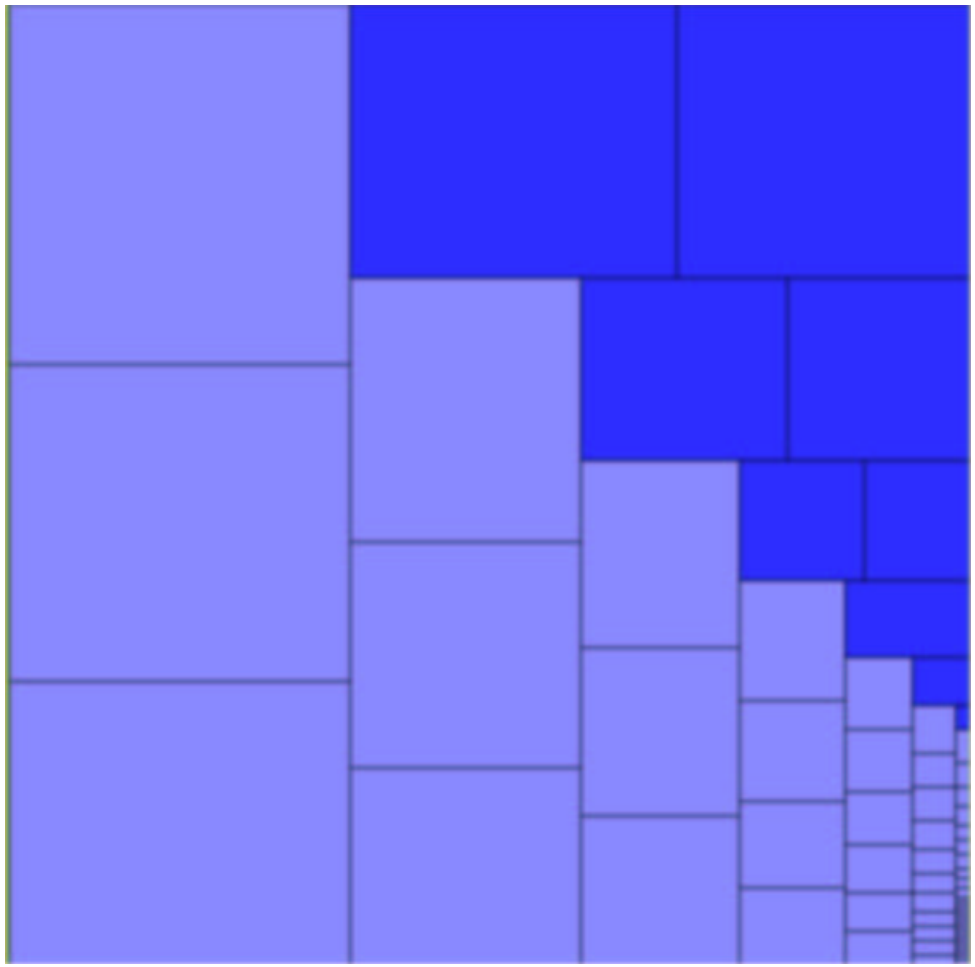
Feedback Guided Loop Scheduling

- The loops iterations are divided into P patches
- Each processor keeps track on time of execution of his patch
- Dividing this time by number of iterations in patch we will get mean load per iteration
- New boundaries are based on equipartitioning of the area under mean load per iteration

Guided 2D

- This algorithm is based on Guided
- Patch size is $\frac{R}{2P}$
- Division provides reasonably square patches trying to keep patches still exponential

Guided 2D

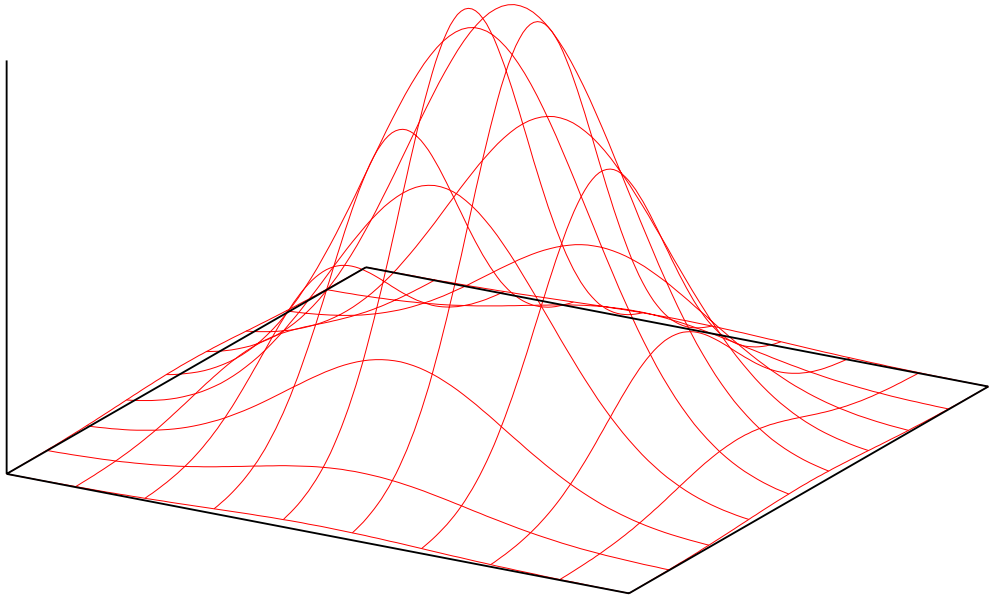


Workload

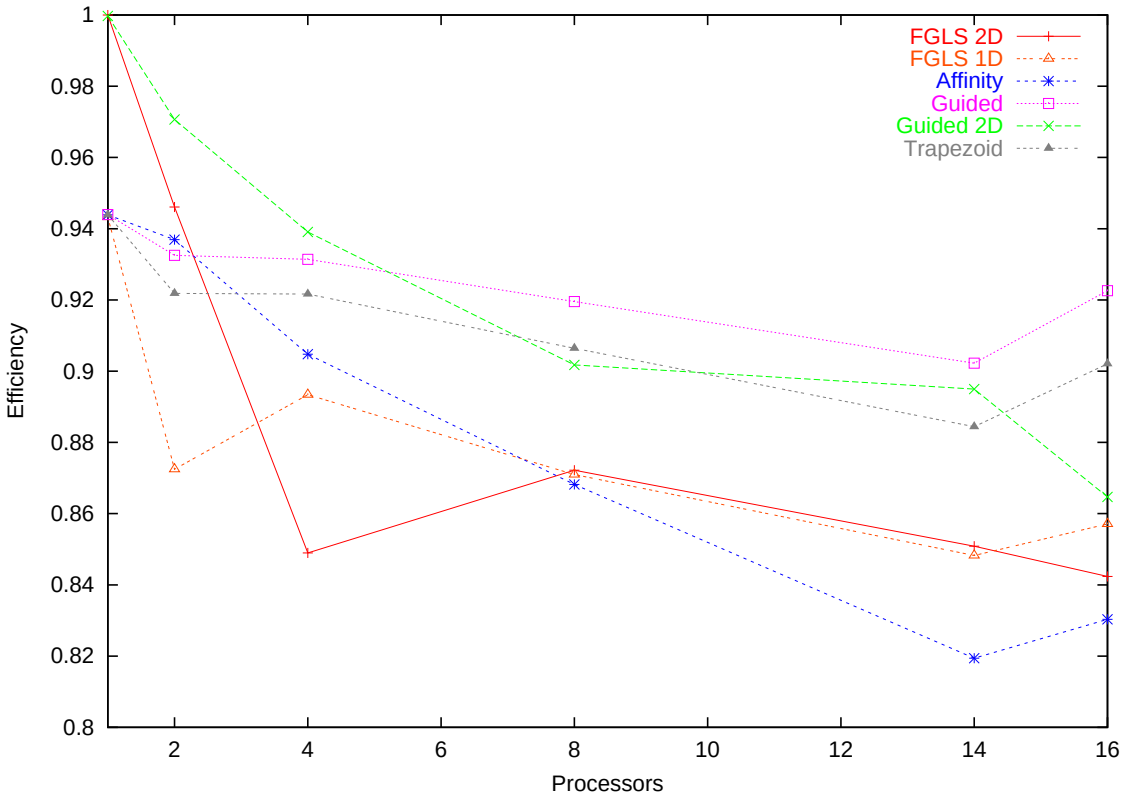
- Workload = communication + imbalance
- Communication load:
  - Each point of area is initialised with some value
  - On every step, each point is mean value of it's neighbours

- Gaussian Load:
- $time = \exp^{-\frac{(x-pos_x)^2+(y-pos_y)^2}{width^2}}$
- $pos_x = center_x + radius * \sin(2 * \pi * iteration/period)$
- $pos_y = center_y + radius * \cos(2 * \pi * iteration/period)$

Gaussian Load

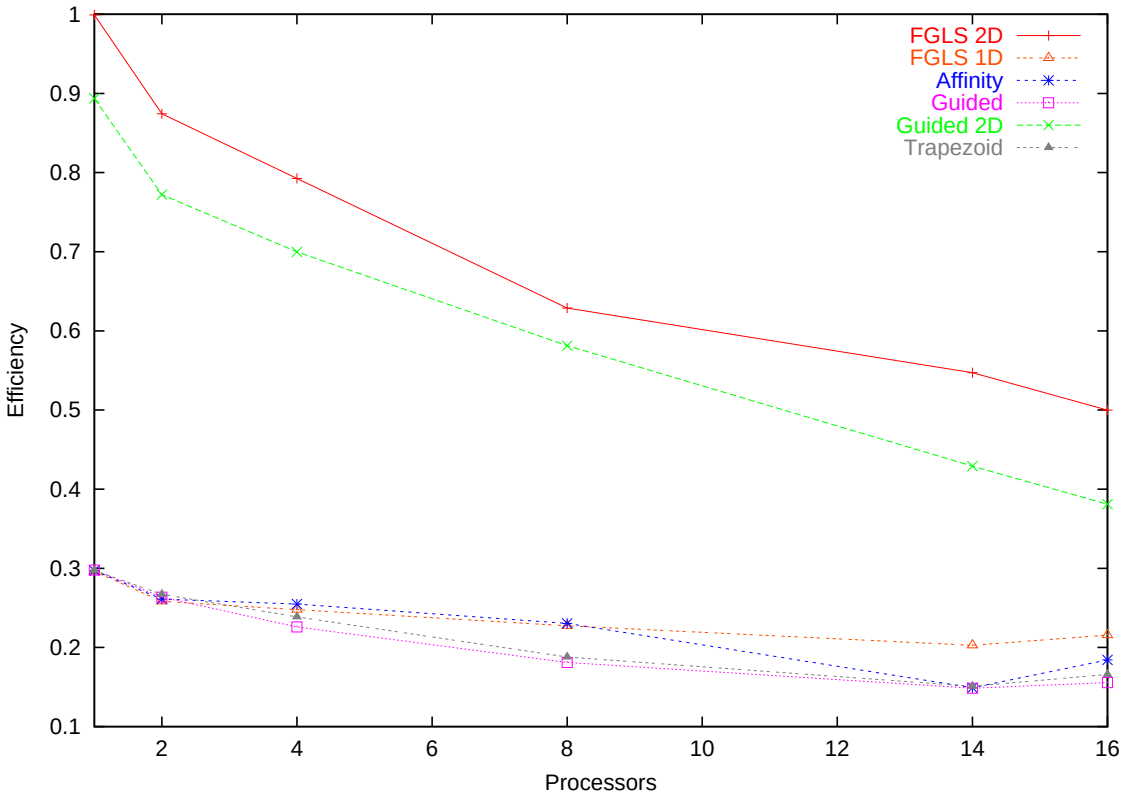


Benchmark results



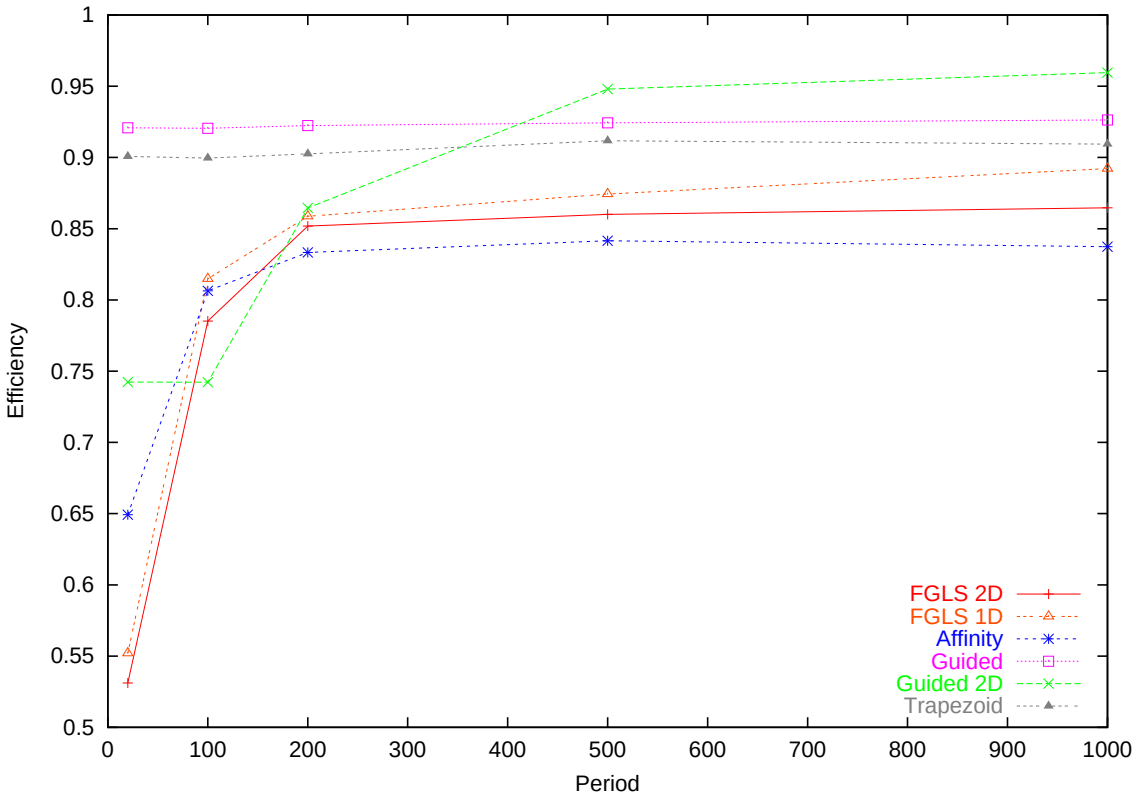
$$Period = 200 Imbalance >> Communication$$

Benchmark results



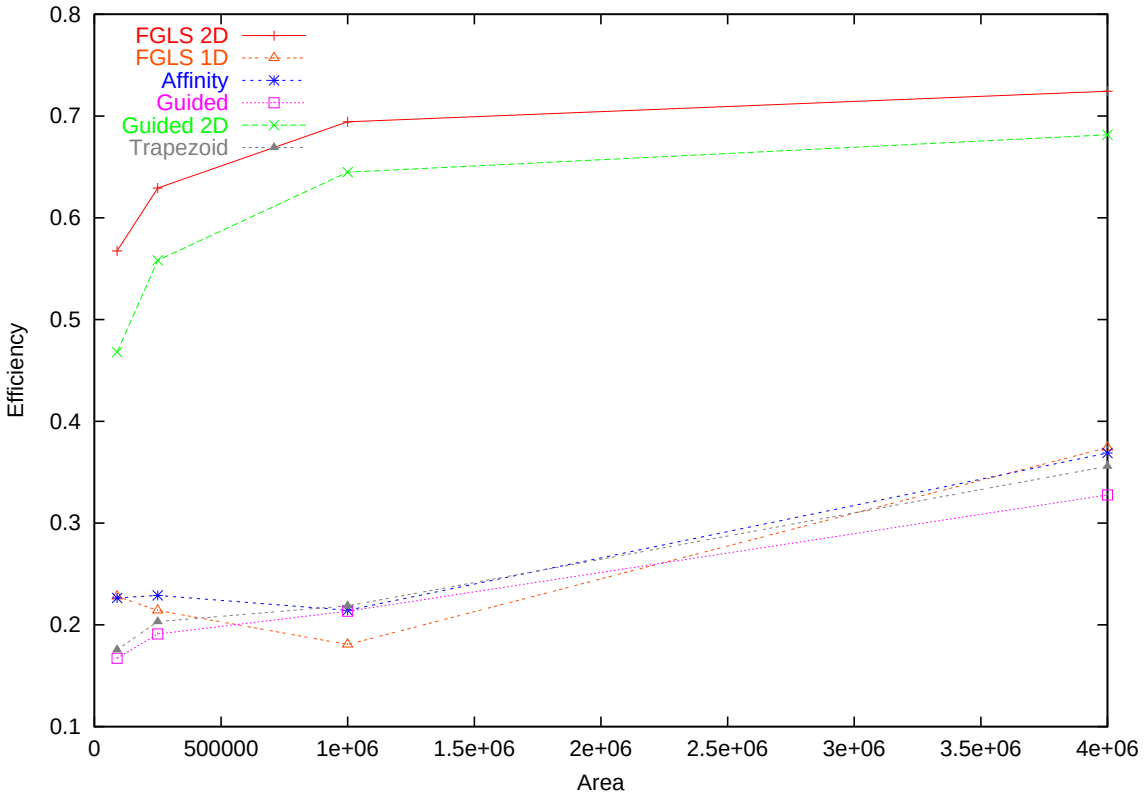
$$Period = 200 Communication \approx Imbalance$$

Benchmark results



$$Processors = 16 Imbalance >> Communication$$

Benchmark results



$$Processors = 16 Communication \approx Imbalance$$