Presentation on “Processor Allocation on Cplant: Achieving General Processor Locality Using One-Dimensional Allocation Strategies” by Leung et al. (leung02a)

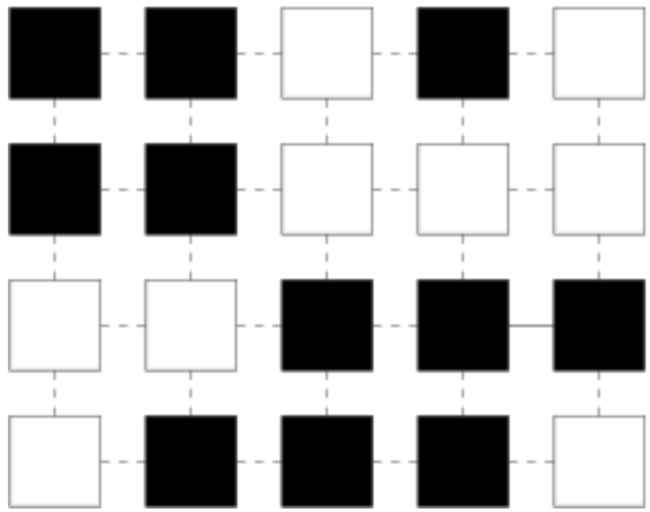
David Bunde

9/18/23

Contributions

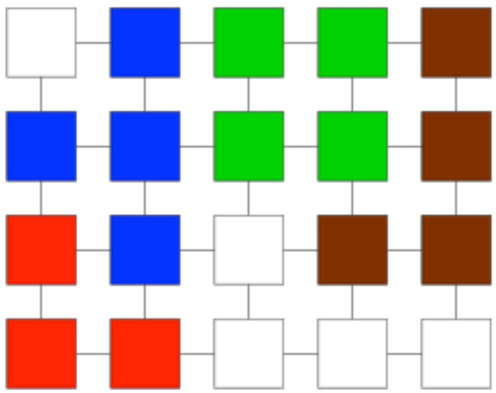
- Evidence that processor allocation matters
- Improved processor allocation scheme for CPlant based on a space-filling curve
- Sum of pairwise distances as a metric for processor allocation

Processor allocation: Where to run

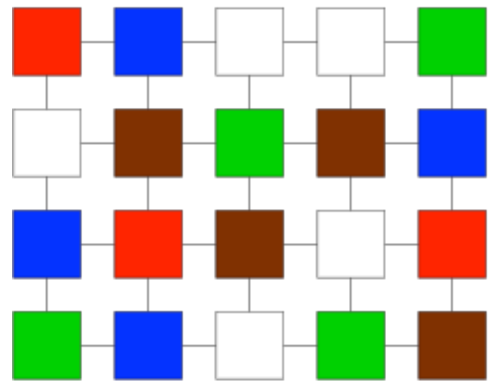


- Which unused (white) processors should system give a 5 node job?

What is a good allocation?

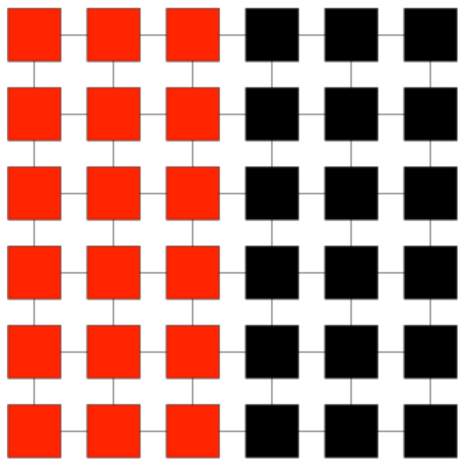


Good allocation

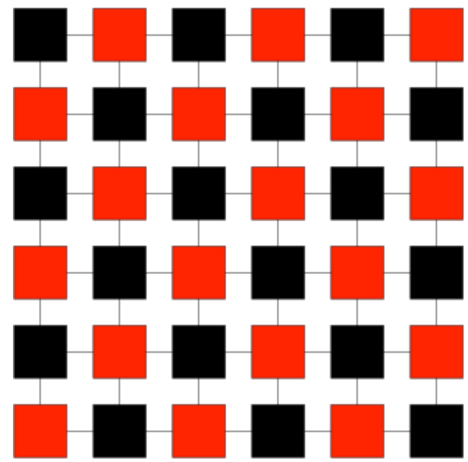


Bad allocation

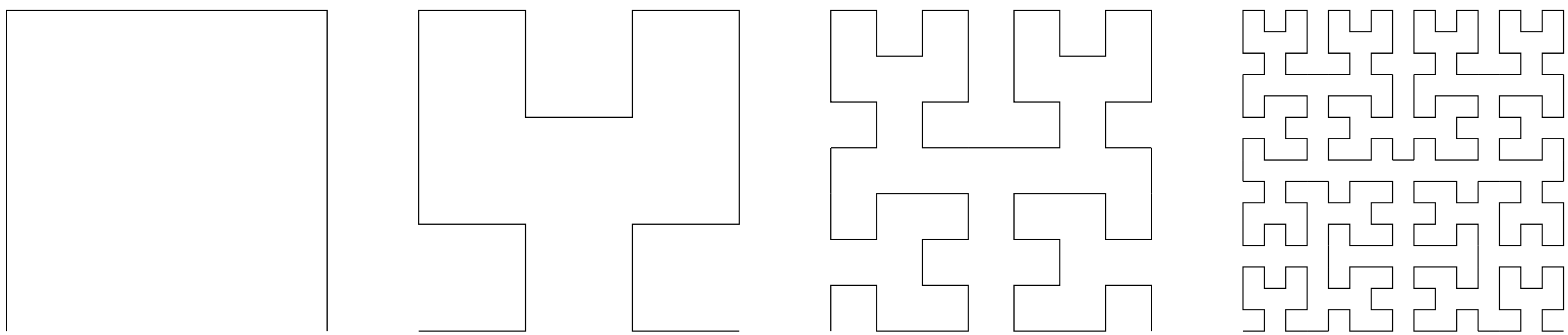
How much does it matter?



runs twice
as fast as



Step 1: Put processors in a good linear order



Hilbert space-filling curve

Step 2: Choose nearby processors in the order

To allocate processors for a single job:

- Free list: assign first processors
- First fit: assign from first interval of free processors (or processors that minimize the range of processors used)
- Best fit: assign from smallest interval of free processors that is big enough (or minimize range)
- Sum of squares: assign from interval that leaves best variety of remaining intervals (or minimize range)

Better allocation reduces running time

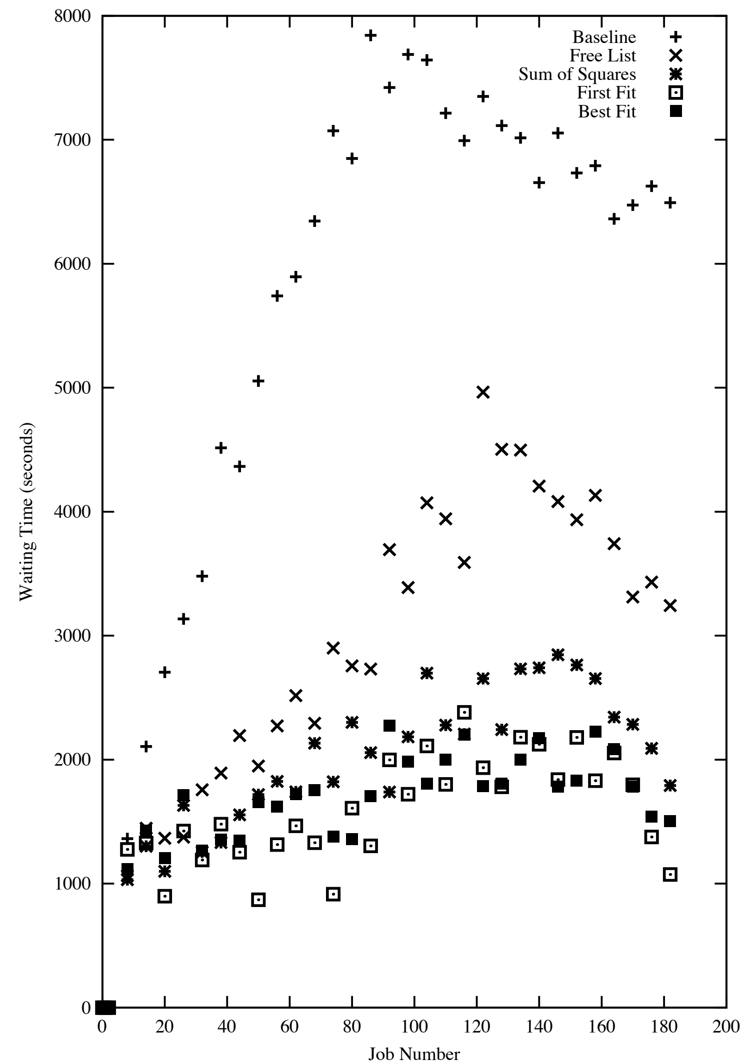


Figure shows large jobs of a trace ordered by submission time (=job number)

Best metric: Sum of pairwise distances

Also looked at: (sum/max/etc)

- Span in linear order
- Span / Job size
- Size of bounding box (3D)
- Sum of bounding box dimensions
- Number of connected components

(But didn't run statistical tests…)