# PARALLEL AND DISTRIBUTED VISION ALGORITHMS

Strandmark P., Kahl F., Schoenemann T.
Lund University, Sweden

## Introduction

We investigate dual decomposition approaches for optimization problems arising in low-level vision. Dual decomposition can be used to:

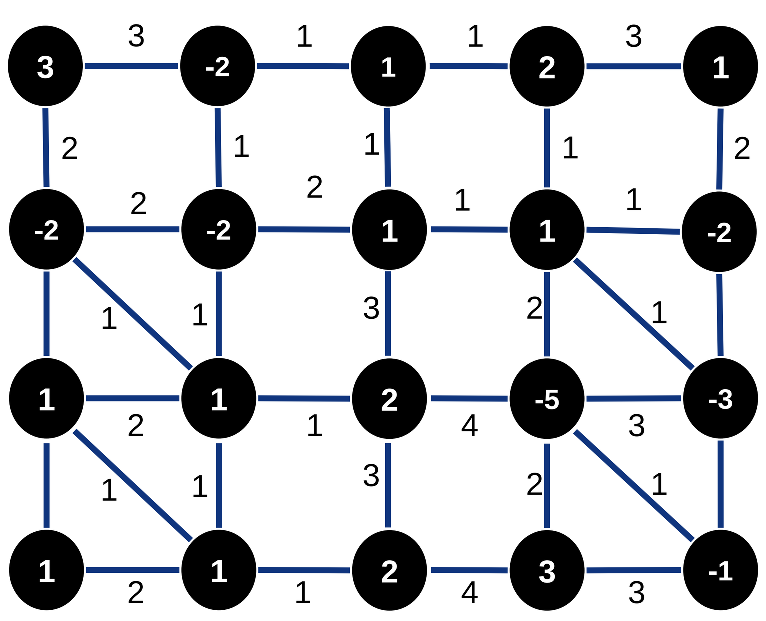
- **Parallelize existing algorithms**
- **Reduce memory requirements**
- **Obtain approximate solutions of hard problems**.

Application considered include graph cut segmentation, curvature regularization and the optimization of general MRFs. We demonstrate that the technique can be useful for desktop computers, graphical processing units and supercomputer clusters.

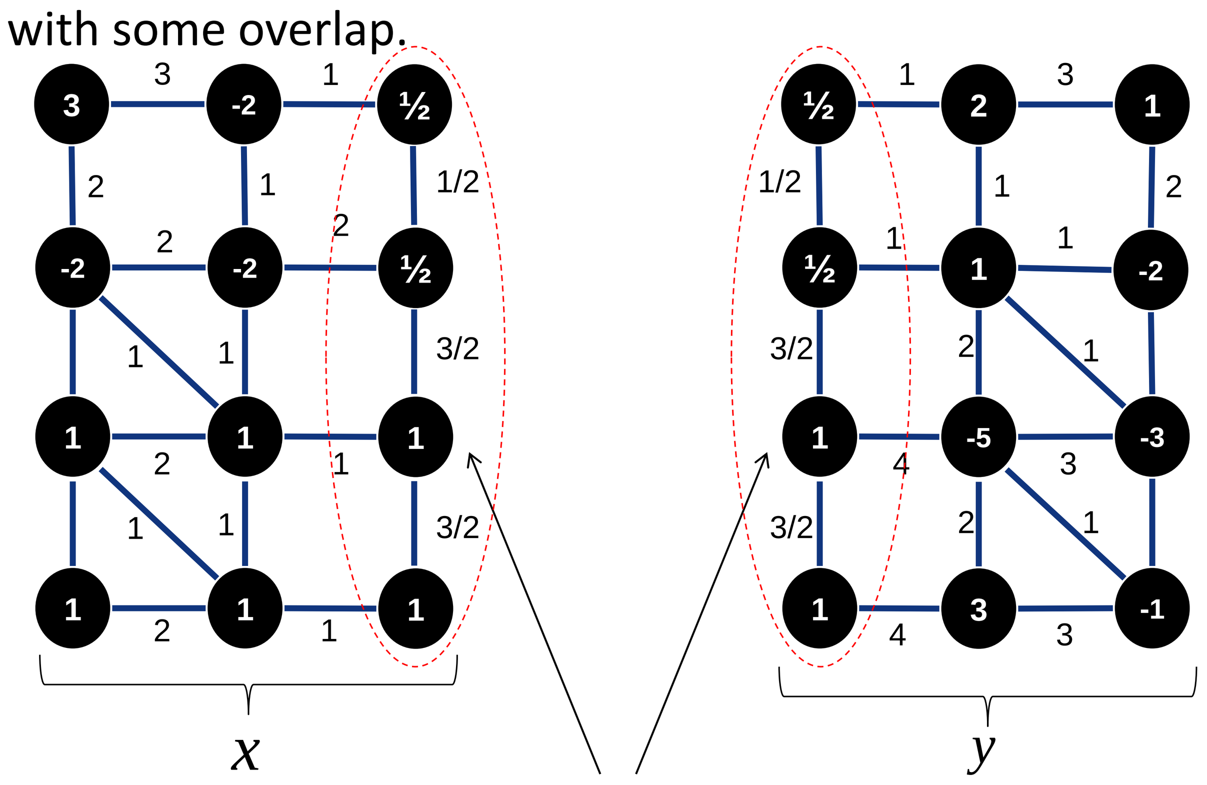
## Parallelizing graph cuts

We begin with a graph:

*Numbers indicate s/t connections and egde costs*



We want to solve this graph in parallel. The trick is to split the graph in two with some overlap.




The nodes that represent the same variable are constrained to be equal with **dual variables**:

$x_i = y_i$ for every i in the overlap.

Each constraint has a dual variable and the dual function to be maximized is:

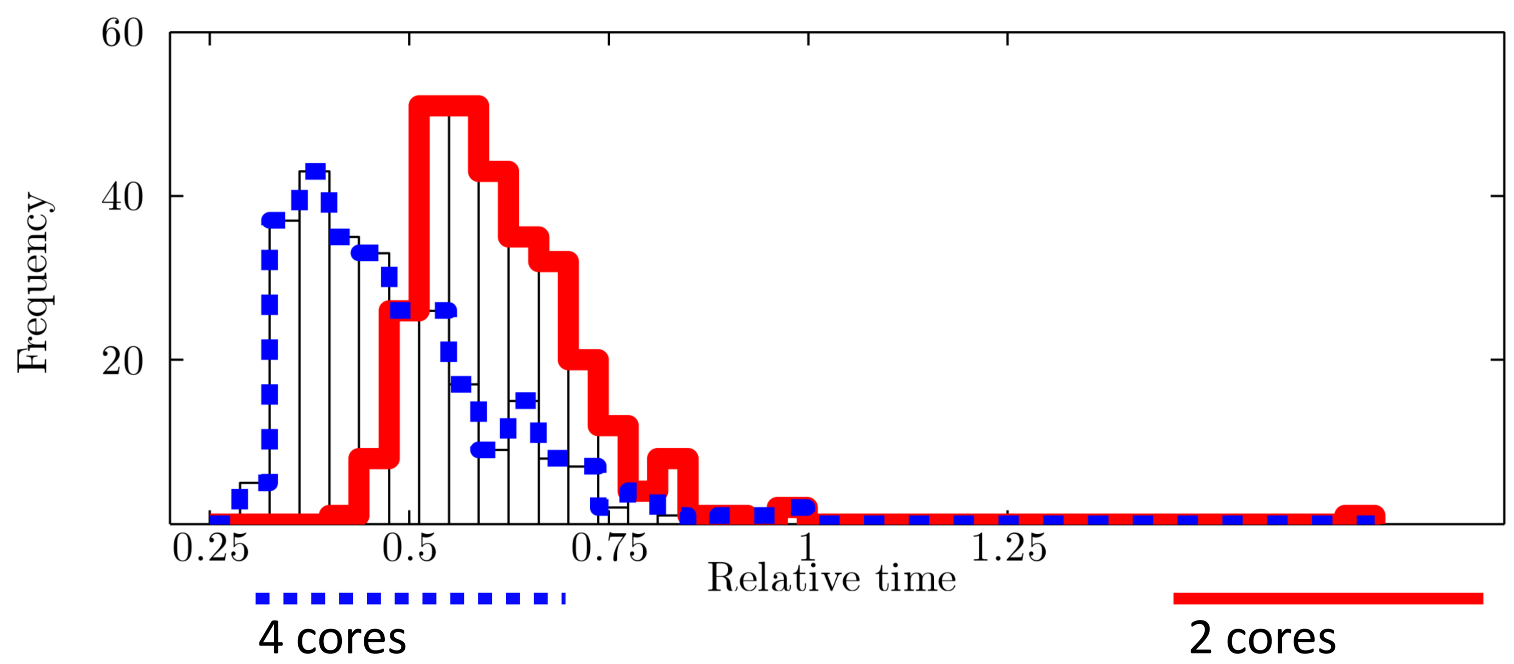
$$g(\lambda) = \min\left( E_{Left}(x) + E_{Right}(y) + \sum_{i \in overlap} \lambda_i (x_i - y_i) \right) =$$

$$\min\left( E_{Left}(x) + \sum_{i \in overlap} \lambda_i (x_i) \right) + \min\left( E_{Right}(y) - \sum_{i \in overlap} \lambda_i (y_i) \right)$$

**Two independent problems!**

We maximize this function with supergradients, solving the two minimum cut problems **in parallel** several times.

Comparison with the common B-K algorithm for 301 images:

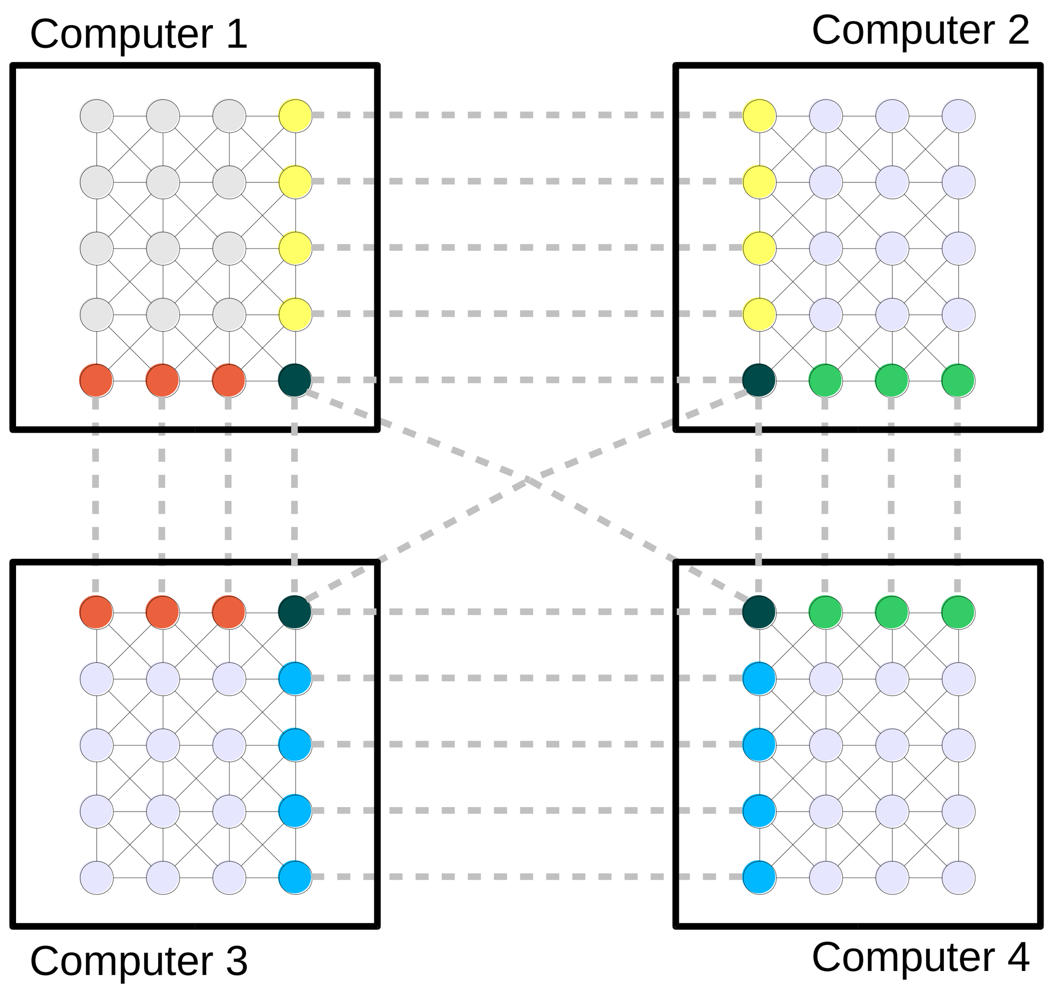



## Reducing memory requirements

Dual decomposition also allows us to save memory, since the graphs to be solved can be allocated to different computers.

This allowed us to solve a graph with **512 × 512 × 2317** vertices and over **3.5 billion** edges.

The graph was divided among 36 machines and solved globally. We are not aware of any previous approaches so solve such huge graphs. The graph above required **131 GB** of primary memory.



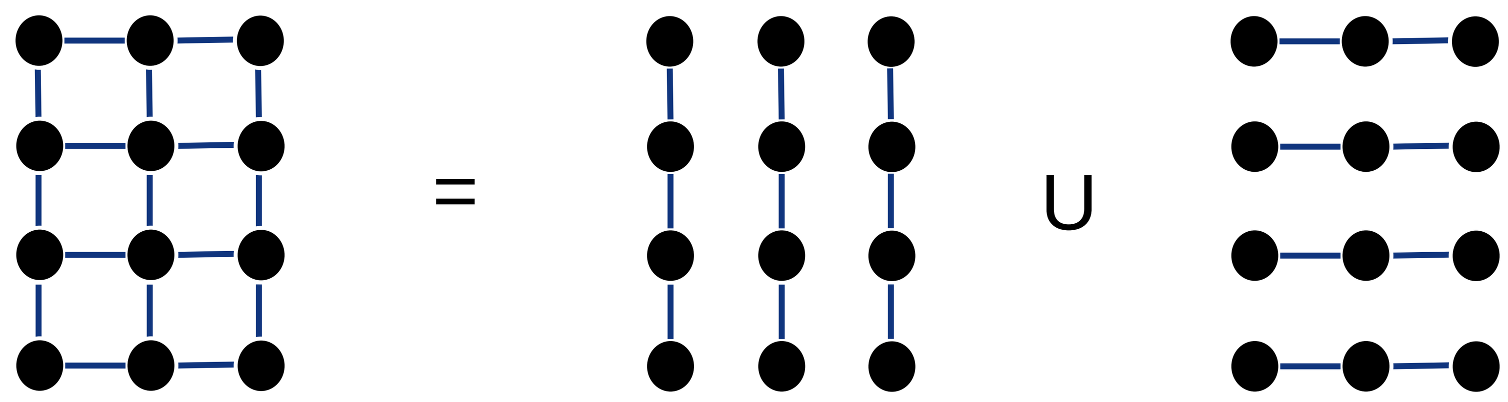

Kahl and Schoenemann demonstrated at ICCV 2009 that problems with curvature regularization can be solved with linear programming. Unfortunately the memory requirements are very large. Our dual decomposition approach applied to those linear programs has allowed us so save memory and solve **larger problems** with **shorter run times** than previously possible. For a small problem, we reduced the peak memory usage from **468** to **279** MB.

## Approximate solutions

Dual decomposition can also be used to obtain approximate solutions to **very hard optimization problems** arising in vision:

$$E(x) = \sum_{i=1}^{n} T_i(x_i) + \sum_{i=1}^{n} \sum_{j \in N(i)} E_{i,j}(x_i, x_j),$$

where x takes a finite number of labels. All functions are arbitrary, so this is a very general formulation. Let us assume that the energy function is defined on a regular grid:




The new problems are one-dimensional and can be solved by dynamic programming with **massive parallelization**.

| Regularization | α-exp. | FastPD | Our (CPU) | Rel. dual gap | Iterations | Our (GPU) |
|---|---|---|---|---|---|---|
| 5×10⁴ | 0.348s | 0.123s | **0.149s** | 0.00092 | 121 | **0.097s** |
| 10⁵ | 0.223s | 0.148s | **0.138s** | 0.00054 | 111 | **0.093s** |
| 10⁶ | 0.500s | 0.318s | **0.086s** | 0.000035 | 46 | **0.043s** |

Comparison between row/column decomposition and two other common methods for a 3-label segmentation problem.

## Conclusions

Dual decomposition is a powerful and simple method to split a large problem into several small subproblems. Often the smaller problems can be made particularly easy to solve and the resulting algorithms is both parallelizable and memory-saving. For convex problems, e.g. minimum cut, global optimality is guaranteed.