Data locality and replica aware virtual cluster embeddings

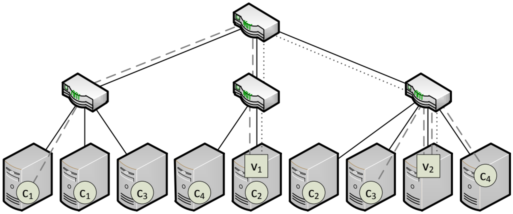
Paolo Costa[1], **Maciej Pacut**[2], Stefan Schmid[3]

[1] Microsoft Research, UK; [2] University of Wroclaw, Poland; [3] TU Berlin & T-Labs, Germany

University of Wroclaw

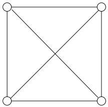
Practical Motivation

Modeling the internals of MapReduce.
Mapping phase, shuffle phase, reduce phase.

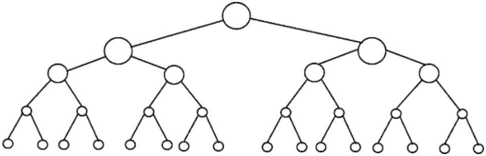


Virtual cluster embedding

Virtual cluster embedding is a task of embedding a clique:

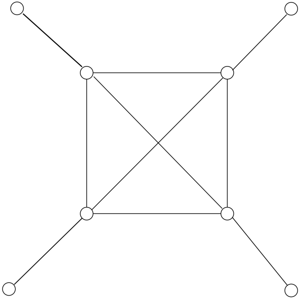


in a leaves of capacitated tree:



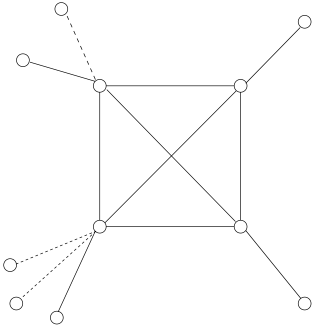
Objective is to perform an embedding that minimizes bandwidth reservations in physical network (tree), respecting bandwidth capacities.

Data locality



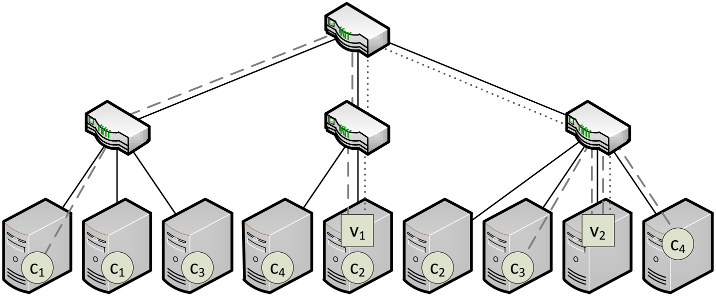
- Objective is to find an assignment of chunks to nodes
- Data can be located in different server
- Transportation is needed
- Embedding a clique + incoming edges
- Non-clique endpoint of incoming edge is fixed

Replica selection



- Data can be stored in redundant way
- Choice of one replica of each chunk type
- Dotted links are replicas that were not chosen to process

Example of chunk and node placement, matching and interconnect



Optimization objective

Objective: embedding that minimizes bandwidth footprint.

$$Objective = \sum_{v \in V} Footprint(v)$$

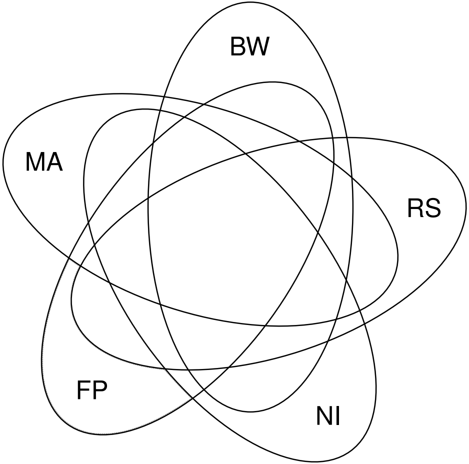
$$Footprint(v) = \underbrace{b_1 \cdot dist(v, \mu(v))}_{\text{transportation}} + \underbrace{\frac{1}{2} \cdot \sum_{v' \in V \setminus \{v\}} b_2 \cdot dist(v, v')}_{\text{inter-connect}}$$

$\mu(v)$ is the chunk assigned to v; the assignment is subject to optimization.

Problem decomposition

1. (FP) Flexible Placement of nodes
2. (RS) Replica Selection
3. (NI) Node Interconnect
4. (MA) Multiple Assignment of chunks to nodes
5. (BW) Bandwidth constraints on physical network links

Problem decomposition - Venn diagram



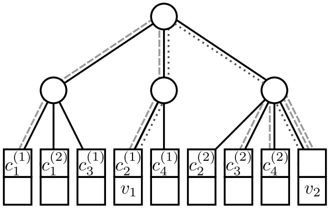
Flexible Placement of nodes, Replica Selection, Node Interconnect, BandWidth, Multiple Assignment of chunks to nodes

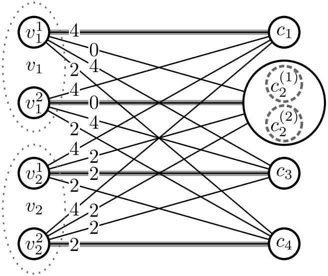
Warm-up - basic model (no extensions)

- (no FP) Node placement is fixed at certain leaves
- (no RS) One replica of each data chunk
- (no BW) Bandwidth is unlimited
- (no MA) Each node processes one data chunk
- (no NI) We just embed the transport of chunks to nodes, without node interconnect (no clique)

solution
=
distance computation + minimum weight perfect matching

Matching approach - replica selection (RS) and multiple assignment (MA)



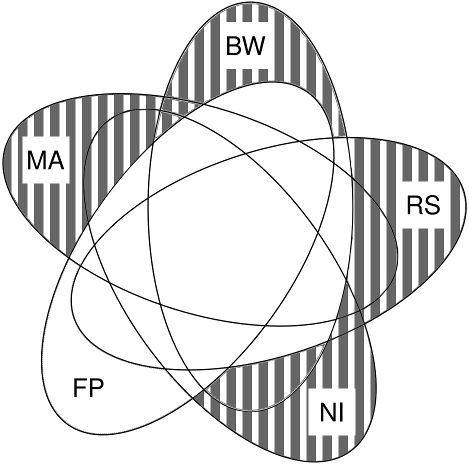


- Each node has to process two chunks $\rightarrow$ the nodes are replicated in the matching representation.
- Two replicas of each chunk type $\rightarrow$ merged into single node with cheapest link
- Minimum weight perfect matching

Local matching and bandwidth

- No Flexible Placement, no Replica Selection
- Local matching on trees is optimal
- Local matching is can be computed in linear time
- We can incorporate bandwith by postprocessing, as if local matching is infeasible, no other matching is feasible.

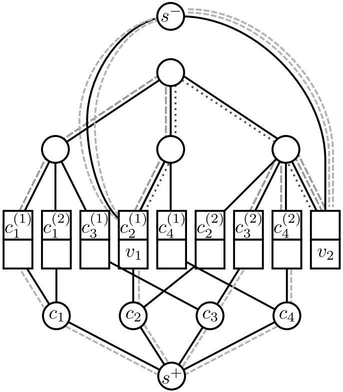
Matching approach - Venn diagram



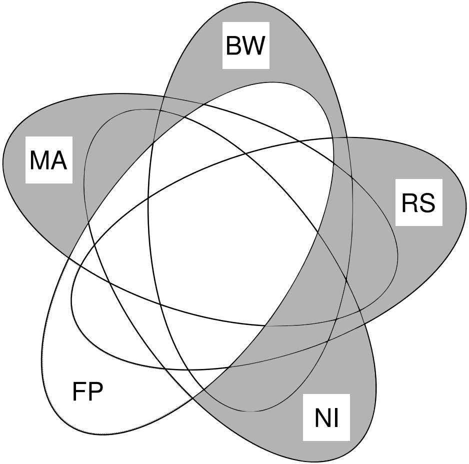
Flexible Placement of nodes, Replica Selection, Node Interconnect, BandWidth, Multiple Assignment of chunks to nodes

Flow approach - replica selection, bandwidth and multiple assignment

- No Flexible Placement
- Artificial graph
- Min-cost flow
- Flow rounding
- Matching by path following
- Example: 2 nodes, 4 chunk types, 2 replicas per type. Dashed line is min-cost flow



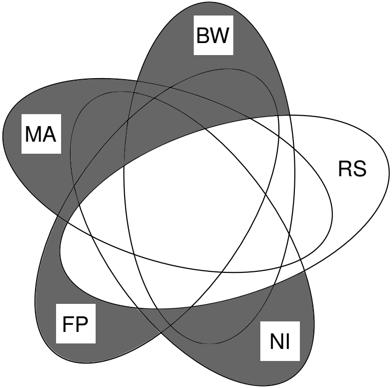
Flow approach - Venn diagram



Flexible Placement of nodes, Replica Selection, Node Interconnect, BandWidth, Multiple Assignment of chunks to nodes

Dynamic program - problem variant introduction

- Embedding of a clique
- Flexible placement
- Bandwidth
- Multiple assignment
- No replica selection



Dynamic program - example

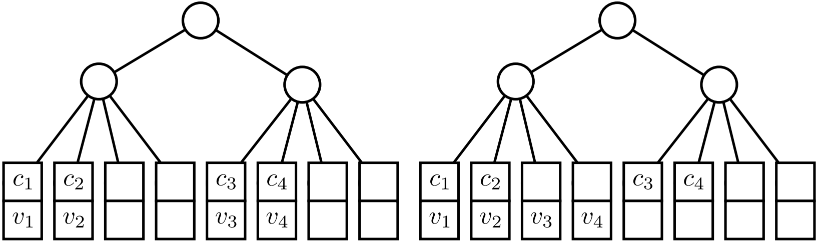


Figure : Two different node placements for the same chunk locations. For $b_1 = b_2$, both solutions have an identical footprint. In other cases, one solution outperforms the other.

Dynamic program

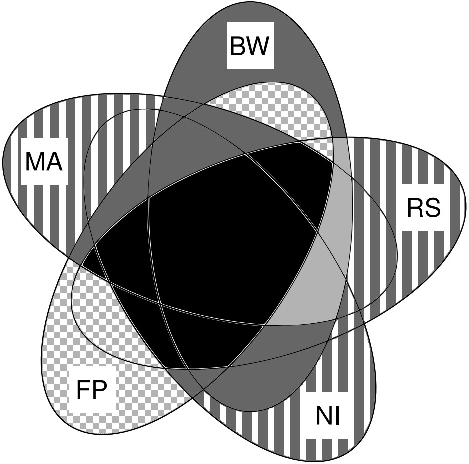
- binarize the tree
- consider all possible number of nodes in every subtree
- computation of local matching
- charge an uplink of each subtree (*bw* function)
- optimal uplink bandwidth depends only on number of nodes in subtree
- follow path of minimas to restore the matching

$$f(T, nodes) =$$

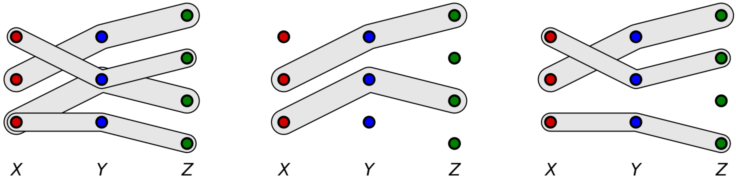
$$\min_{0 \le right \le nodes} \{f(T_{left}, nodes - r) + f(T_{right}, r)\} + bw(T, nodes)$$

Hardness results

Remaining variants of the problem are either trivial or NP-complete.



Introduction to 3D Perfect Matching



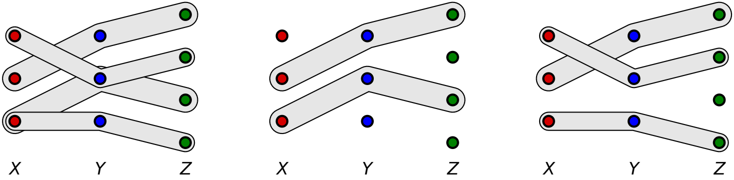
- Input: sets X, Y, Z and set of triples
- Goal: choose subset of triples that covers every element of $X \cup Y \cup Z$ exactly once

Reduction intuitions

- For every element in $X \cup Y \cup Z$, we create a chunk type.
- (3D Perfect Matching) cover each element exactly once

  $$\Longleftrightarrow$$

  (Virtual Cluster) each chunk type must be processed exactly once
- Encoding of triple as a gadget with three leaves
- To turn the optimization problem into a decision problem, we will use a cost threshold Th.

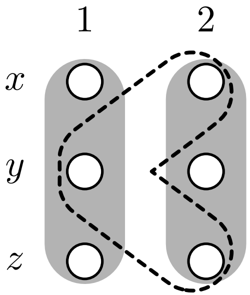


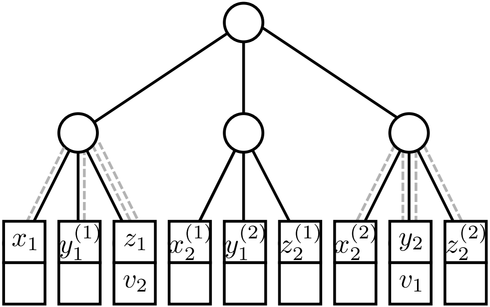
3D Perfect Matching = Exact Cover ∩ 3-Set Cover

Exact cover - to avoid processing the chunk type multiple times.
3-Set Cover - to set threshold upfront.

Hardness of multiple assignment

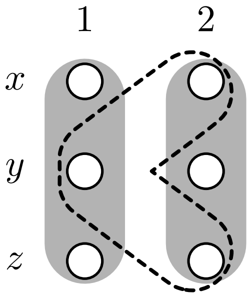
- Flexible Placement, Replica Selection, Multiple Assignment
- Solution = the grey triples
- The dashed triple is not used for the solution
- Each node processes 3 chunks (MA)
- Threshold = $4 \cdot k$ (to prevent transportation among gadgets)

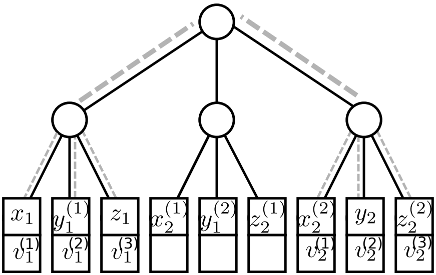




Hardness of interconnect

- Flexible Placement, Replica Selection, Node Interconnect
- Size of clique $= 3 \cdot k$.
- Threshold $= 18 \cdot k^2 - 12 \cdot k$ (to avoid spreading nodes across more than k gadgets)





NP-hardness - conclusion

$$
\begin{array}{ccc}
\mathbf{RS+FP+NI} & \Longrightarrow & \mathrm{RS+FP+NI+BW} \\
\Downarrow & & \Downarrow \\
\mathrm{RS+MA+FP+NI} & \Longrightarrow & \mathrm{RS+MA+FP+NI+BW} \\
\Uparrow & & \Uparrow \\
\mathbf{RS+MA+FP} & \Longrightarrow & \mathrm{RA+MA+FP+BW}
\end{array}
$$

Figure : The NP-hardness of 2 variants, implies that 4 other variants are also NP-hard.

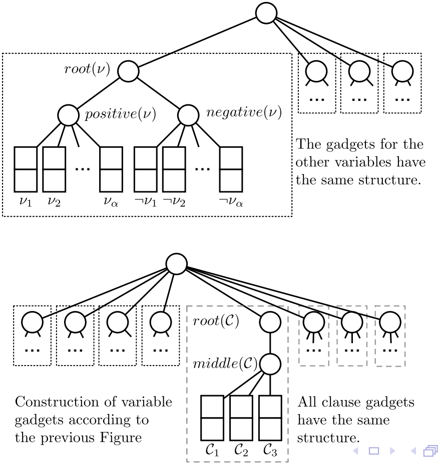
Summary in tabular form

NP-hard	5 combinations	RS + MA + FP + NI + BW
	4 combinations	RS + MA + FP + NI; RS + MA + FP + BW; RS + FP + NI + BW
	3 combinations	RS + MA + FP; RS + FP + NI
Flow	4 combinations	RS + MA + NI + BW
	3 combinations	RS + NI + BW; RS + MA + BW
	2 combinations	RS + BW
DP	4 combinations	MA + FP + NI + BW
	3 combinations	MA + FP + NI; MA + FP + BW; FP + NI + BW
	2 combinations	MA + FP; FP + NI;
Matching	3 combinations	RS + MA + NI; MA + NI + BW
	2 combinations	RS + MA; RS + NI; MA + NI; MA + BW; NI + BW
	1 combination	RS; MA; NI; BW
0 Cost	3 combinations	RS + FP + BW
	2 combinations	RS + FP; FP + BW
	1 combinations	FP

Table : Fastest algorithms for different respective problem variants.

Further results

Further results: replication of factor 2 is enough for the problem to remain NP-hard in scenerio with node interconnect. Again, using small-diameter networks. Reduction is from 3SAT.



Thank you!

Thank you!