ASP Multi-Level Global Placement (MLGP)

ML Cluster Growth and Flattening/Filtering

Nikos Sketopoulos, Pavlos Aimoniotis and Christos Sotiriou

1

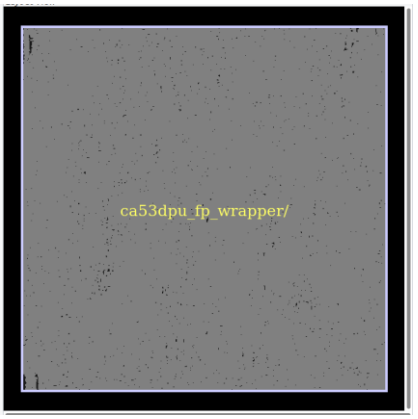
ASP Multi-Level Clustering for Global Placement

- Multi-level (ML) Global Placement Aims:
  - Flat GP Speed Up
    - Divide-and-Conquer Strategy
    - Parallelization of Independent GP (Sub-)Problems
  - Reduce High Fanout Nets Issues *(N^2 entries in Adjacency Matrix)*
- ML Clusters Growth
  - Combination of Edge Coarsening and Hyper-Edge Coarsening (hMetis)
  - Based on Net Fanout, Area Normalised Net Fanout or Affinity
  - Goal is for Clusters of the same Level to be as area balanced as possible
- ASP ML Clustering Algorithm Goals and Global Variables:
  - Area Balance Between Levels N, N+1 Clusters (`mlupperboundratio`)
  - Area Balance Between Same Level Clusters (`mllevelupperboundratio`)
  - Reduction in Number Clustered Objects per Level (`mllevelreduction`)
  - Clusters at Final Level; Terminator (`mlfinallevelclustersnum`)

2

Global Placement

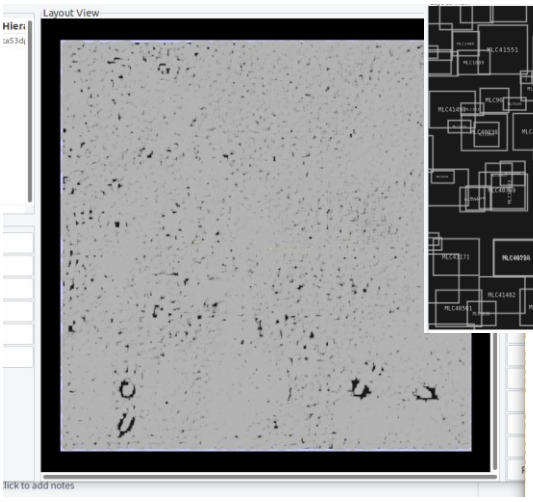
Flat Global Placement

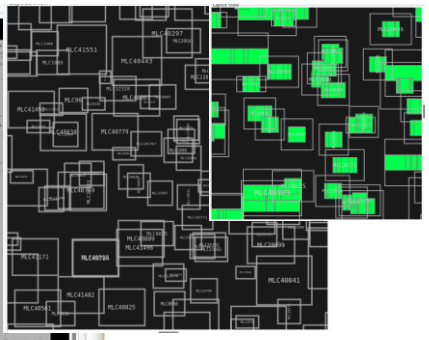


Partitioned Global Placement

3

Global Placement





4

Original EC ML Cluster Growth (Original Code)

```
Function:
level = 0;
S = sort_nets_by_connectivity();

do
  {
    grow_clusters(level, S);
    level++;
  } while (get_clusters_num(level) > FINALNUM)
```

Completely grow each cluster based on highest fanout net;
Then, move on to next net and corresponding cluster, in decreasing fanout order.

```
Function: cluster_grow(level, S)
cluster_id = 0;
result = 0;
foreach net i in S
  {
    if (cluster_id = get_clusters_num(level-1)/REDUCTION)
      break;

    foreach object obj in i
      {
        if (is_clustered(obj))
          continue;
        result = check_area(obj, cluster_id, average(level-1)*RATIO);
        if (result)
          add_obj(i, cluster_id);
      }
    if (result)
      cluster_id++;
  }
```

Unbalanced Clusters at the Same Level

5

Improved ML Cluster Growth (Improved Code)

```
Function: cluster_grow(level, S)
cluster_id = 0;
result = 0;
min_area = initial_value;

foreach net i in S
  {
    foreach object obj in i
      {
        if (is_clustered(obj))
          continue;
        result = check_area(obj, cluster_id, min_area*LRATIO);
        if (result)
          {
            add_obj(i, cluster_id);
            min_area = get_min_cluster_area();
            break;
          }
      }

    if (result)
      cluster_id++;

    if (cluster_id = get_clusters_num(level-1)/REDUCTION)
      {
        if (!result)
            faults++
        if (faults = get_clusters_num(level-1)/REDUCTION)
          break;
        i = get_candidate_net();
      }
  }
```

Initial_value = ???

```
Function: clustering()
level = 0;
S = sort_nets_by_connectivity();

do
  {
    grow_clusters(level, S);
    level++;
  } while (get_objects_num(level) > FINALNUM)
```

Clustering takes place until current level objects is < than the required level number; objects are either components, existing mlclusters or structural clusters.

More Uniform Clusters at the Same Level

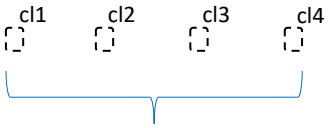
6

ML Cluster Growth

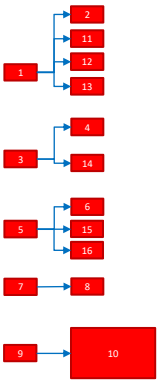
- Idea of Multiple Cluster Growing it to grow all current level clusters by adding one object at the time
  - except for the first move

Example:

- Specify the number of the clusters



4 clusters must created/grown in this level



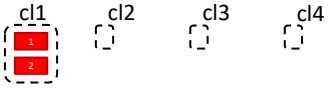
7

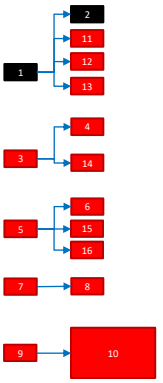
ML Cluster Growth

- Idea of Multiple Cluster Growing it to grow all current level clusters by adding one object at the time
  - except for the first move

Example: Round 1

- Add the 1st pair of objects of highest fanout net (or affinity)
- Lock used objects



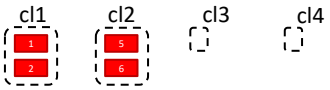


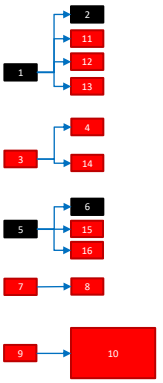
8

ML Cluster Growth

- Idea of Multiple Cluster Growing it to grow all current level clusters by adding one object at the time
  - except for the first move

Example: Round 1
- Add the 1st two objects of the 2nd highest fanout net
- Lock them



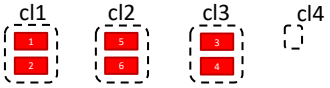


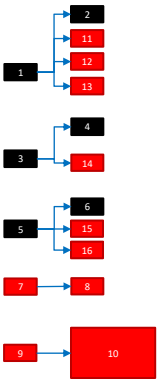
9

ML Cluster Growth

- Idea of Multiple Cluster Growing it to grow all current level clusters by adding one object at the time
  - except for the first move

Example: Round 1
- Add the 1st two objects of the 3rd highest fanout net
- Lock them



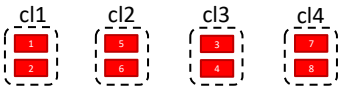


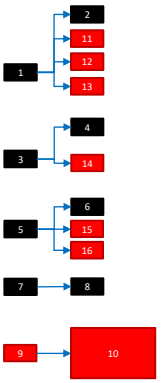
10

ML Cluster Growth

- Idea of Multiple Cluster Growing it to grow all current level clusters by adding one object at the time
  - except for the first move

Example: Round 1
- Add the 1st two objects of the 4th highest fanout net
- Lock these objects



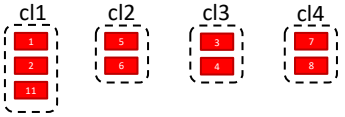


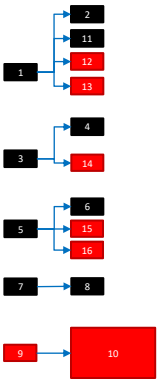
11

ML Cluster Growth

- Idea of Multiple Cluster Growing it to grow all current level clusters by adding one object at the time
  - except for the first move

Example: Round 2
- Add the next object of the 1st highest fanout net
- Lock it



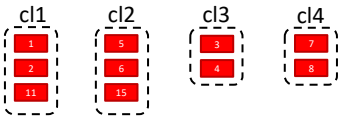


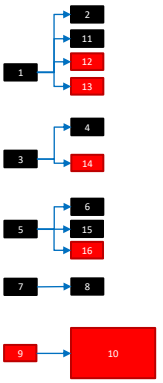
12

ML Cluster Growth

- Idea of Multiple Cluster Growing it to grow all current level clusters by adding one object at the time
  - except for the first move

Example: Round 2
- Add the next object of the 2nd highest fanout net
- Lock it



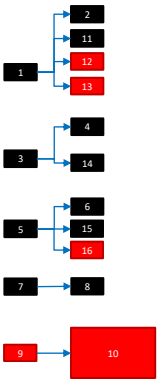


13

ML Cluster Growth

- Idea of Multiple Cluster Growing it to grow all current level clusters by adding one object at the time
  - except for the first move

Example: Round 2
- Add the next object of the 3rd highest fanout net
- Lock it

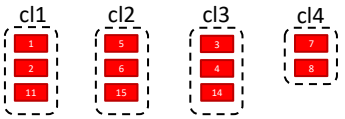


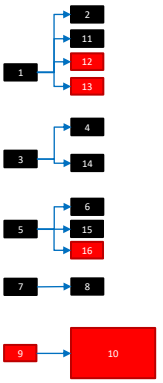
14

ML Cluster Growth

- Idea of Multiple Cluster Growing it to grow all current level clusters by adding one object at the time
  - except for the first move

Example: Round 2
- Add the next object of the 3rd highest fanout net
- Lock it





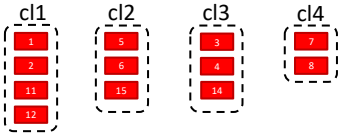
If we want ML clusters to be area, growth must stop at round 2; otherwise, *e.g.* if area balance is 2, we continue until area balance constraint is satisfied.

15

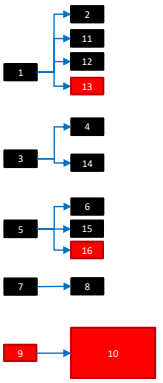
ML Cluster Growth

- Idea of Multiple Cluster Growing it to grow all current level clusters by adding one object at the time
  - except for the first move

Example: Round 3
- Add the next object of the 1st highest fanout net
- Lock it



Area balance is 2

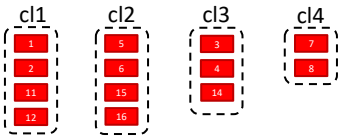


16

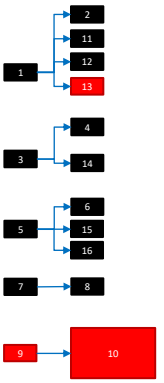
ML Cluster Growth

- Idea of Multiple Cluster Growing it to grow all current level clusters by adding one object at the time
  - except for the first move

Example: Round 3
- Add the next object of the 2nd highest fanout net
- Lock it



Area balance is 2

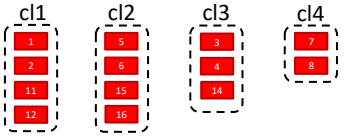


17

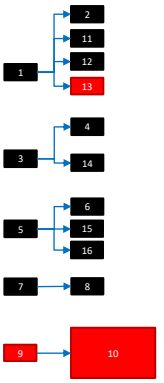
ML Cluster Growth

- The idea of Parallel Clusters Growing it to grow all clusters of the current level, by adding one object at the time (except the first grow).

Example: Round 3
- Add the next object of the 2nd highest fanout net
- Lock this it

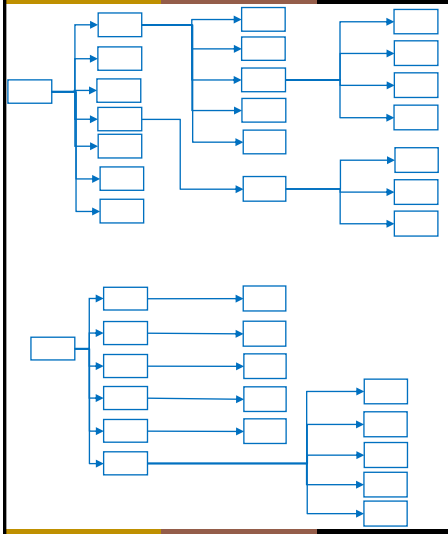


Object "13" will NOT added to ML Cluster "cl1" due to the area balance criterion, as the area of cl1 is twice that of smaller ML Cluster (cl4)



18

Example 1



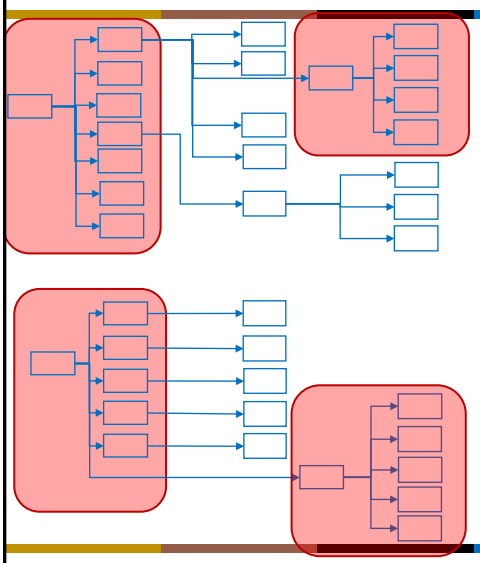
38 objects
12 nets

reduction = 10
finalnum = 2
ratio = 2
levelratio = 2

38/10 = 4 clusters
10 objects @ each cluster

19

Example 1 – Level 1



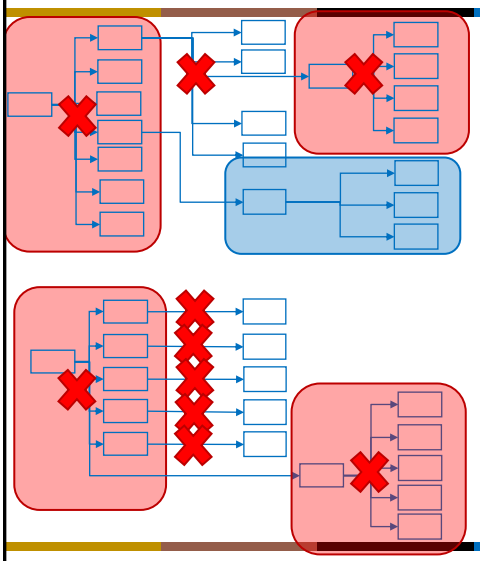
17 objects
8 nets

reduction = 10
finalnum = 2
ratio = 2
levelratio = 2

17/10 = 2 clusters
10 objects @ each cluster

20

Example 1 – Level 2 with Small Initial Value



17 objects
8 nets

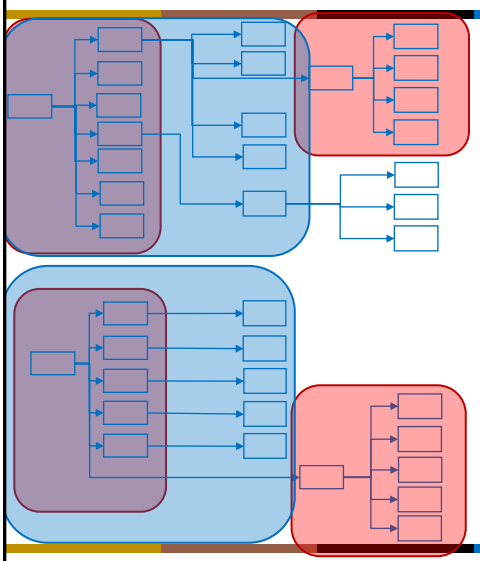
reduction = 10
finalnum = 2
ratio = 2
levelratio = 2

17/10 = 2 clusters
10 objects @ each cluster

Small initial balance value,
e.g. 2 x Average Object Area of L1
(2x(38/17) = 4.5)

21

Example 1 – Level 2 with Large Initial Value



17 objects
8 nets

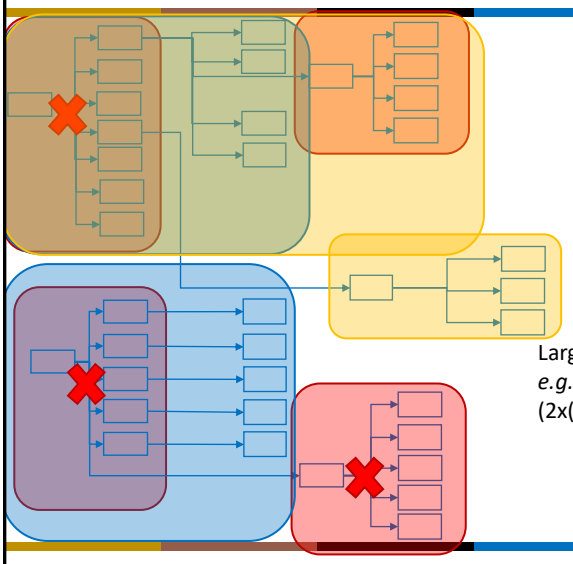
reduction = 10
finalnum = 2
ratio = 2
levelratio = 2

17/10 = 2 clusters
10 objects @ each cluster

Large initial balance value,
e.g. 2 x Average Clustered Area at L1
(2x(25/6) = 12.5)

22

Example 1 – Level 3 with Large Initial Value



12 objects
7 nets

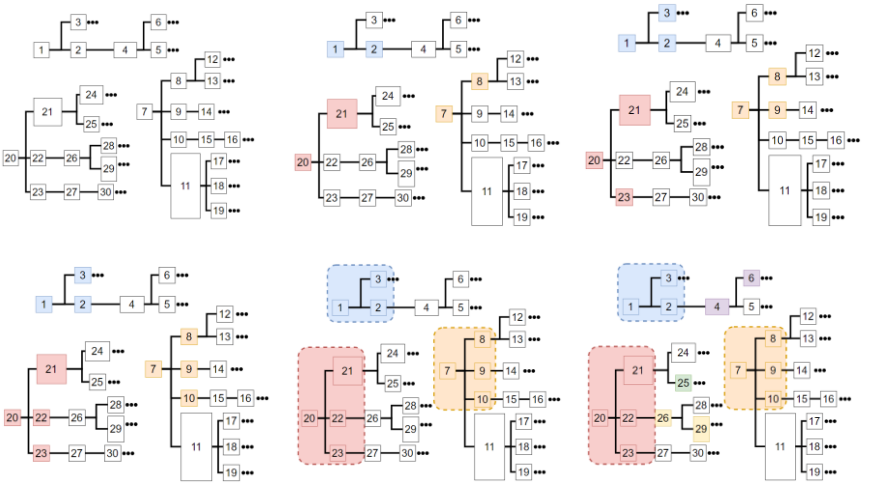
reduction = 10
finalnum = 2
ratio = 2
levelratio = 2

12/10 = 2 clusters
10 objects @ each cluster

Large initial balance value,
e.g. 2 x Average Clustered Area at L2
(2x(29/3) = 19)

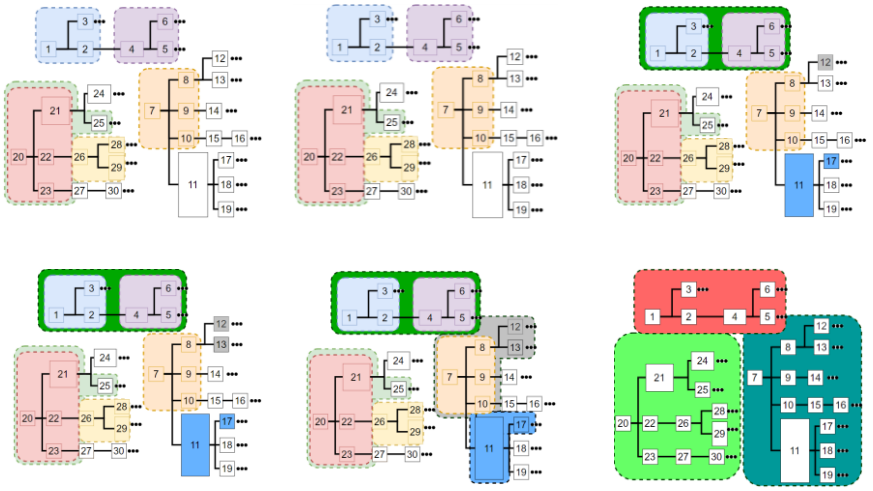
23

Example 2



24

Example 2



25

Problems

```
Function: cluster_grow(level, S)
cluster_id = 0;
result = 0;
min_area = initial_value;

foreach net i in S
  {
    foreach object obj in i
      {
        if (is_clustered(obj))
          continue;
        result = check_area(obj, cluster_id,
min_area*LRATIO);
        if (result)
          {
            add_obj(i, cluster_id);
            min_area = get_min_cluster_area();
            break;
          }
      }

    if (result)
      cluster_id++;

    if (cluster_id = get_clusters_num(level-
1)/REDUCTION)
      {
        if (!result)
            faults++
        if (faults = get_clusters_num(level-
1)/REDUCTION)
          break;
        i = get_candidate_net();
      }
```

```
Function: clustering()
level = 0;
S = sort_nets_by_connectivity();

do
  {
    grow_clusters(level, S);
    level++;
  } while (get_objects_num(level) >
FINALNUM)
```

- Have to create a lot of levels in order to group cells of differnt nets
- During those levels the creted clusters will be unbalanced
- The iterative nature of the algorithm and the number of levels leads to increased execution time overhead

26

Alternative Algorithm

Algorithm 1 MultiNet Clustering Algorithm Top-Level

Input: Netlist ($Cells$, $Nets$), Sorting Order (O), Final Objects Number (FO), Level Reduction (LR), Upper Area Bound Ratio (UBR), Level Upper Area Bound Ratio ($LUBR$), Minumum Clusters per Level (MCL).
Output: Set of Clusters per Level, up to a computed Maximum Level, satisfying input parameters.

1: SN = `sort_nets`($Nets$, O); // sort nets based on specified order O //
2: l = 1; // level 0 is cell level //
3: **repeat**
4: $|clusters(l)|$ = `grow_mlclusters`($Netlist$, SN, LR, UBR, $LUBR$, MCL, l); // grow current level clusters //
5: $l = l + 1$;
6: **if** ($|$clusters(l)$|$ $< MCL$) **then**
7: **break;** // cluster creation saturated at current level //
8: **end if**
9: **until** (`|objects(`l`)|` $< FO$); // clustering exit condition //
10: `flatten_mlclusters`(*clusters per l*, *maxlevel*); // post clustering Flattening step to guarantee MNM //

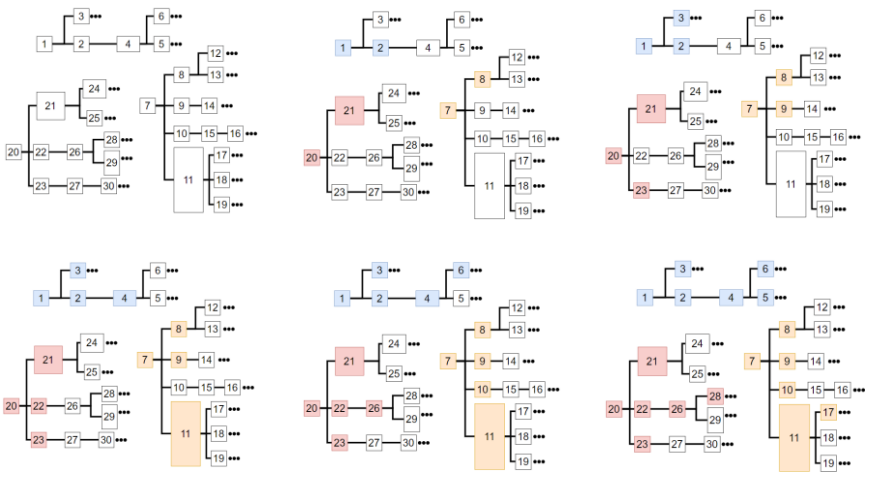
Algorithm 2 Multilevel Clustering Core Algorithm, *i.e.* `grow_mlclusters()` function of Algorithm 1.

Input: Netlist ($Cells$, $Nets$), Sorted Nets (SN), Level_Reduction (LR), Upper Area Bound Ratio (UBR), Level Upper Area Bound Ratio ($LUBR$), Minimum Clusters per Level (MCL), Current Level (l)
Output: Set of Clusters at current Level, ensuring area balance between them and aiming for their number to be $\geq (\frac{1}{LR})\times$ previous level Clusters.

1: // $clusterednets$ = list of nets corresponding to current level clusters //
2: // $mlclusternets[j]$ = additional nets associated to cluster of seed net j, related to cluster cell net contents //
3: $phase$ = 1;
4: **for** (j in Sorted Nets SN) **do**
5: **if** ($phase$ == 2) **then**
6: j = *next clustered net in clusterednets*; // clusters fill-in phase //
7: **end if**
8: (*netcovered*, *candidate*) = `get_net_candidate_object`(j, l);
9: **if** ($netcovered$ == 1) **then**
10: // must add additional nets to mlcluster of net j // $mlclusternets[j] = mlclusternets[j] \cup$ `get_additional_mlcluster_nets`(j, l);
11: **else**
12: // check area bounds, and if satisfied, insert candidate into mlcluster of net j // result = `check_area_and_insert`(j, *candidate*, UBR, $LUBR$, l); // result indicates whether area bounds are satisfied //
13: **if** ($result$ == 1) **then**
14: **if** ($phase$ == 1) **then**
15: $clusterednets = clusterednets \cup j$;
16: **else if** ($phase$ == 2) **then**
17: $failednets$ = 0; // reset failed nets count upon successful clustering //
18: **end if**
19: **else if** ($result$ == 0) **then**
20: // clustering candidate of net j failed //
21: **if** ($phase$ == 2) **then**
22: $failednets = failednets + 1$;
23: **if** ($failednets == |clusterednets|$) **then**
24: **return clusters;** // all candidate nets failed; clustering Phase II ends //
25: **end if**
26: **end if**
27: **end if**
28: **end if**
29: **if** ($|objects(l)| \geq (\frac{|objects(l-1)|}{LR})$) **then**
30: $phase$ = 2; // move from phase 1 to phase 2, i.e. fill-in phase //
31: **end if**
32: **end for**

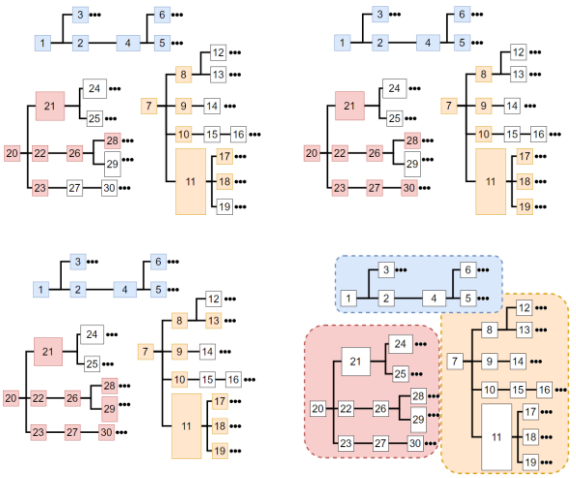
27

Alternative approach example 1



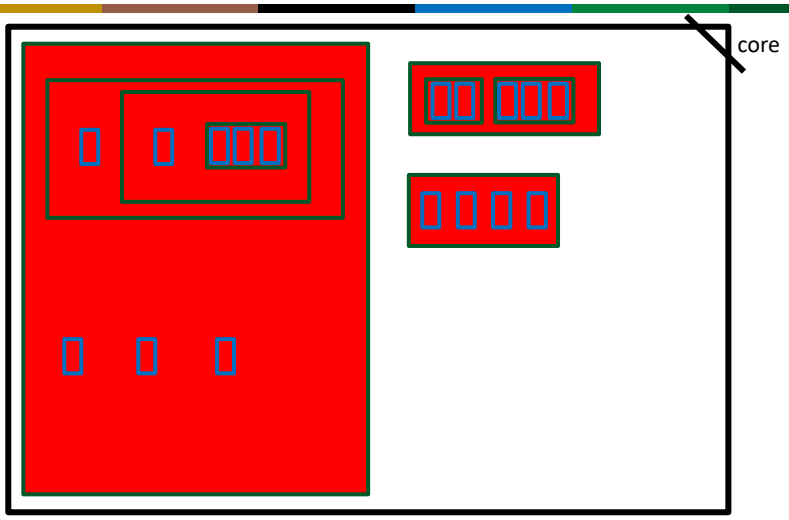
28

Alternative approach example 1



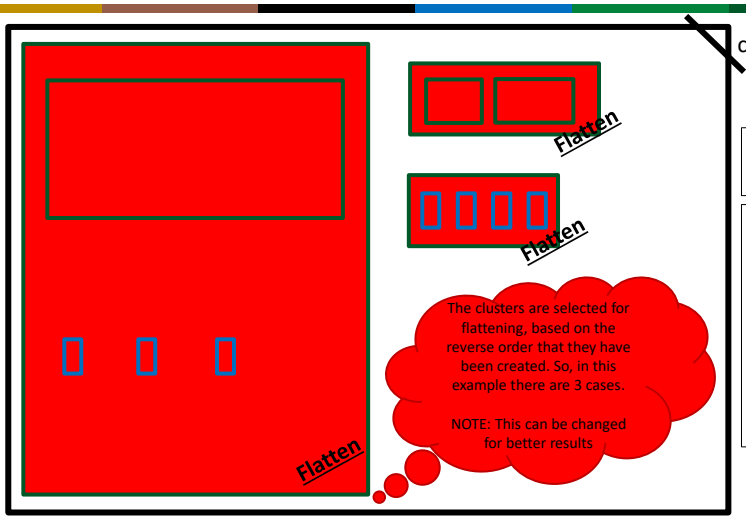
29

Clusters Flattening



31

Clusters Flattening – Top-Down

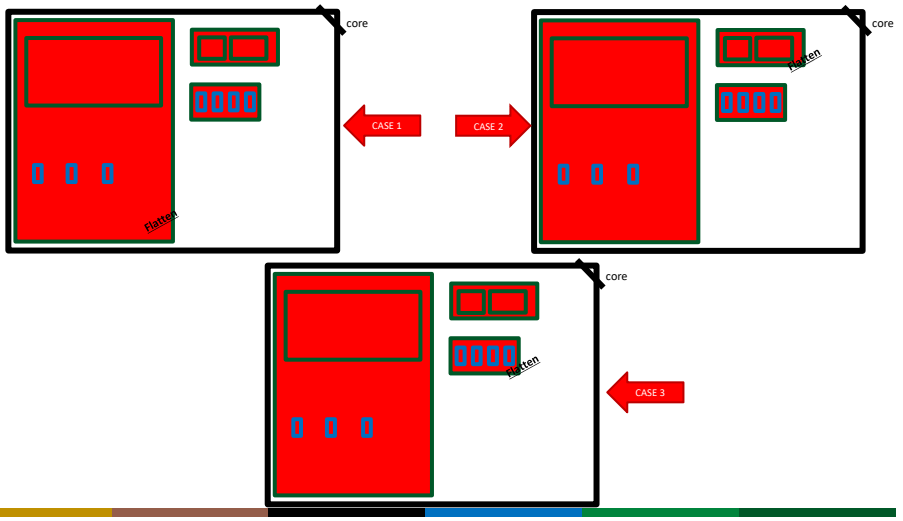


3 Clusters at Top Level

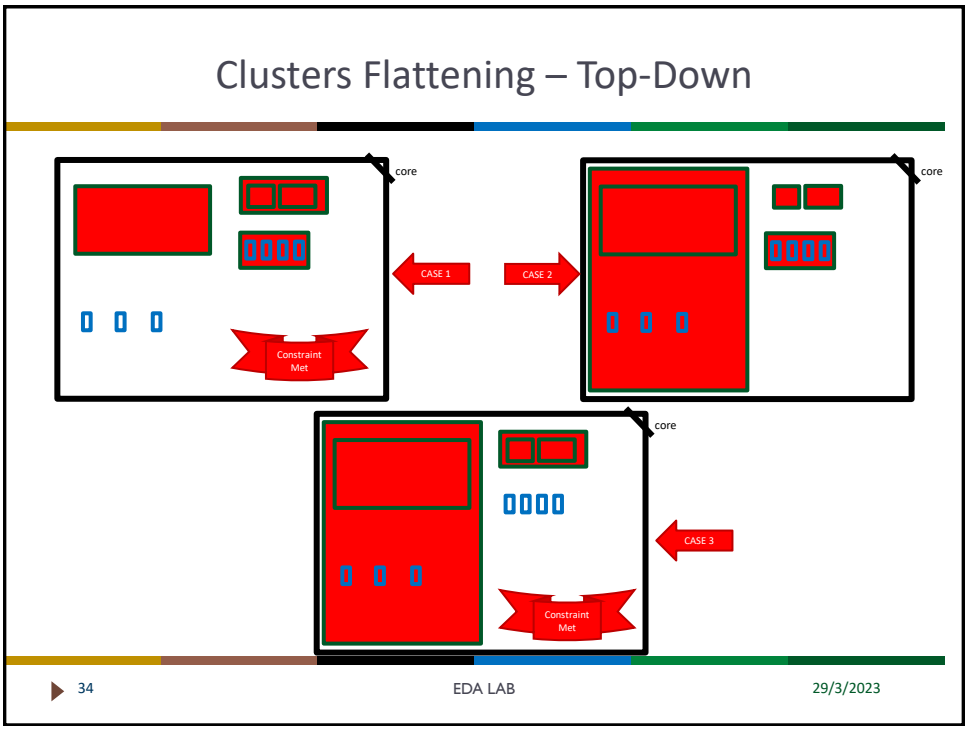
Flatten Clusters Until there are at least N, *e.g.* 5, members at each cluster scope

32

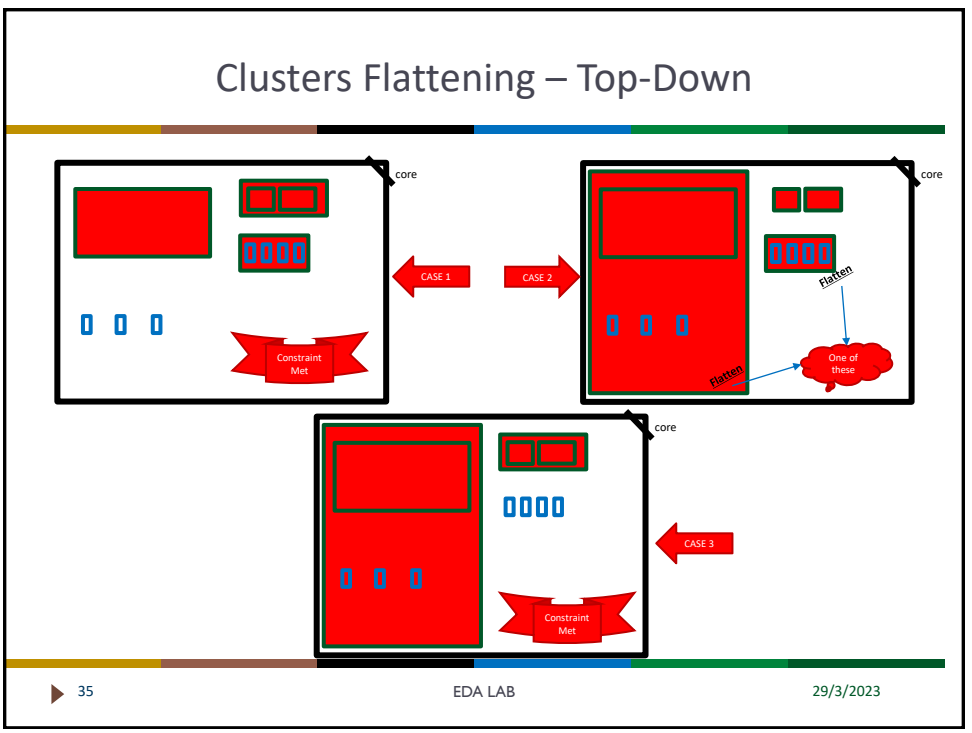
Clusters Flattening – Top-Down



33

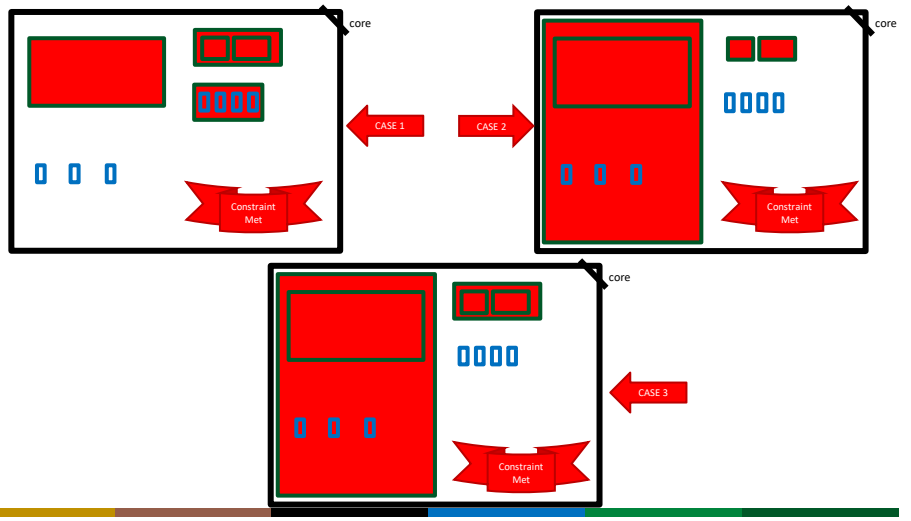

Clusters Flattening – Top-Down
core
CASE 1
CASE 2
core
Constraint Met
core
CASE 3
Constraint Met
34
EDA LAB
29/3/2023

34


Clusters Flattening – Top-Down
core
CASE 1
CASE 2
core
Flatten
Flatten
One of these
Constraint Met
core
CASE 3
Constraint Met
35
EDA LAB
29/3/2023

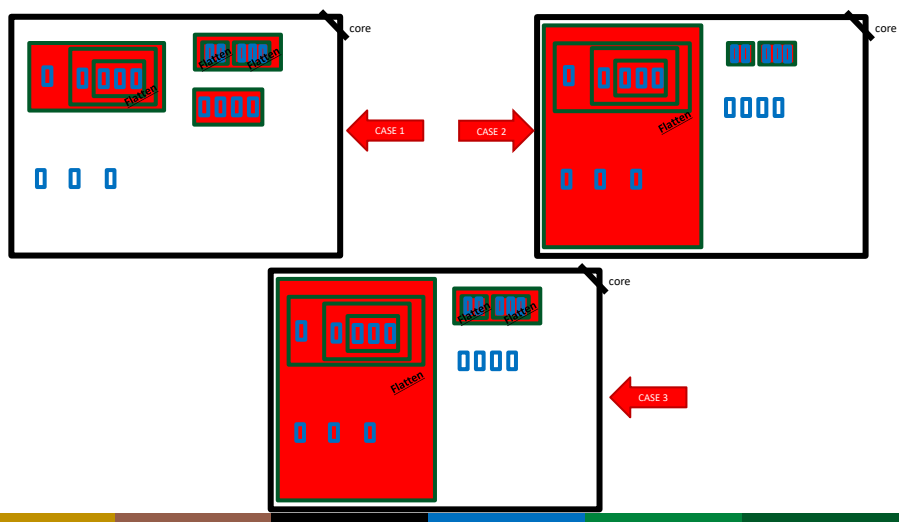
35

Clusters Flattening – Top-Down



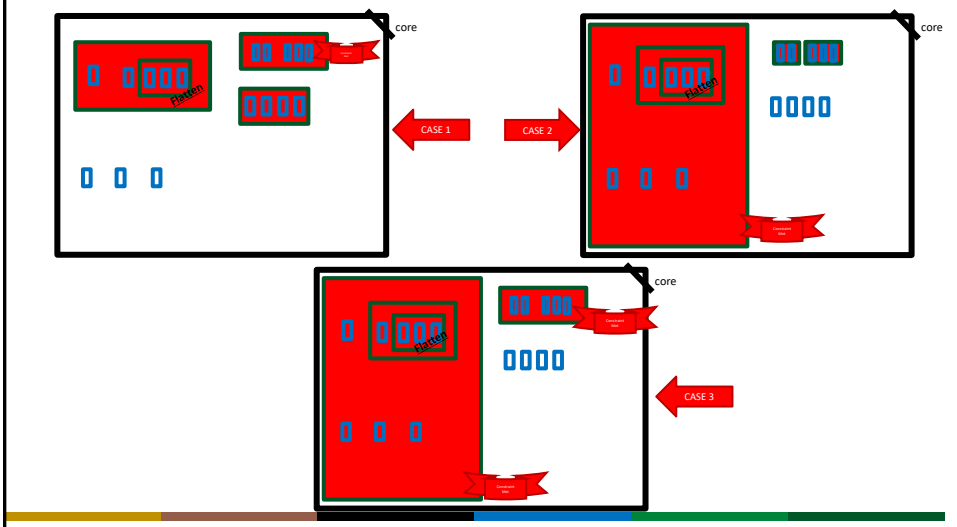
36

Clusters Flattening – Top-Down



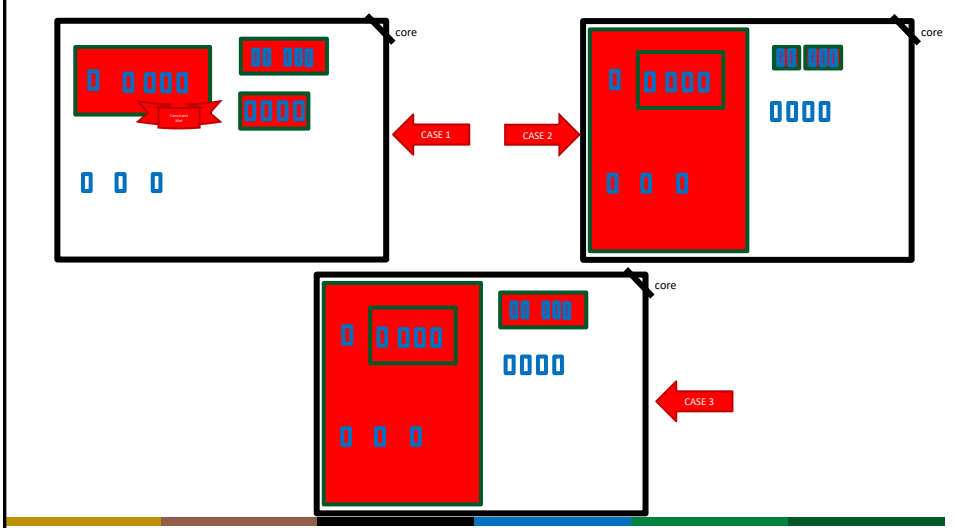
37

Clusters Flattening – Top-Down



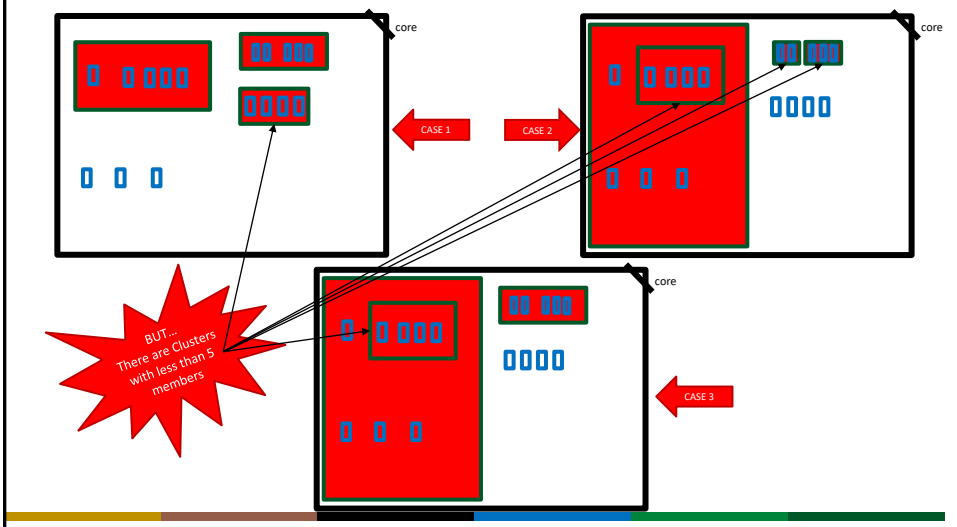
38

Clusters Flattening – Top-Down



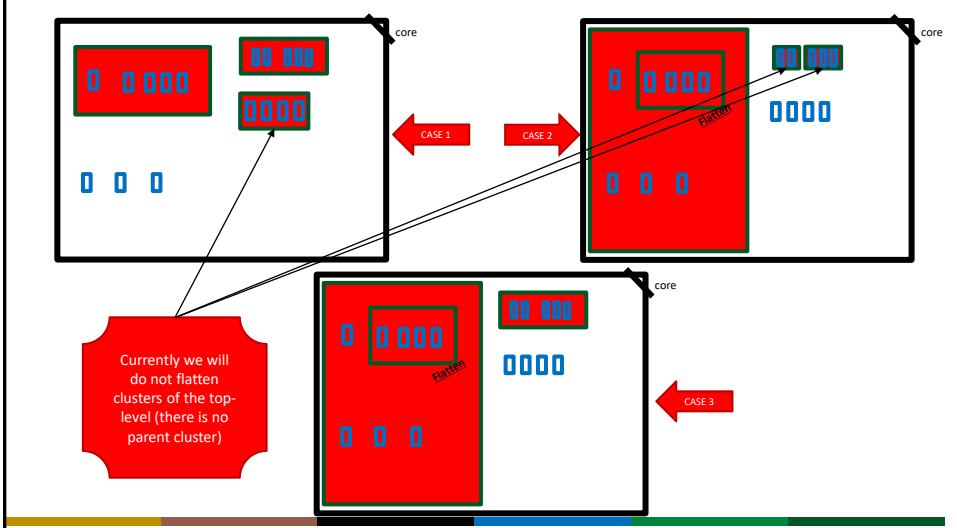
39

Clusters Flattening – Top-Down



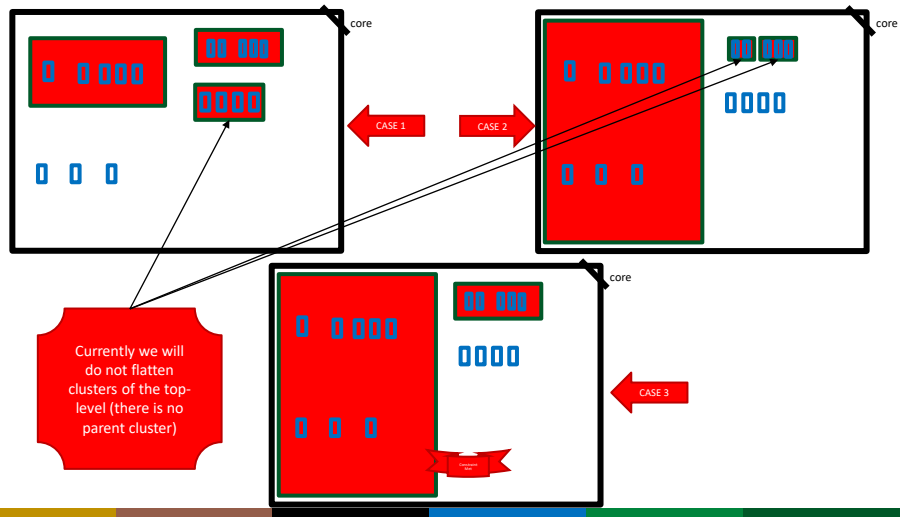
40

Clusters Flattening – Top-Down



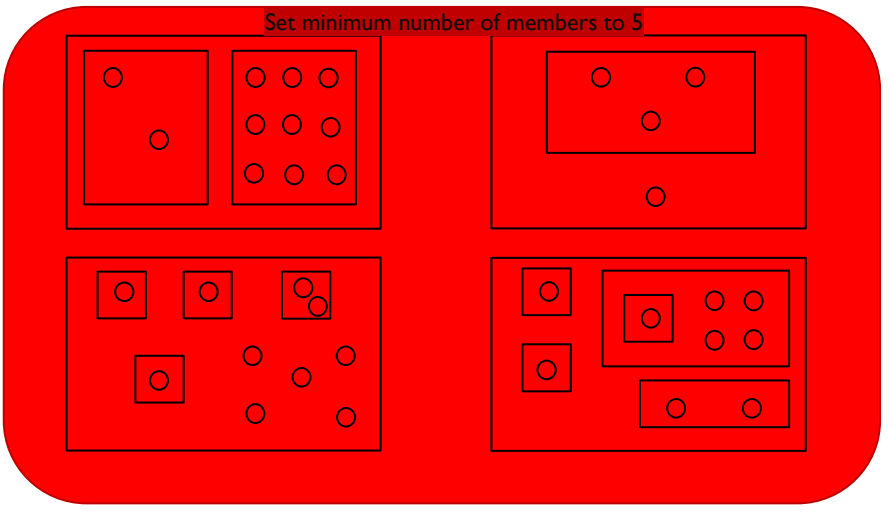
41

Clusters Flattening – Top-Down



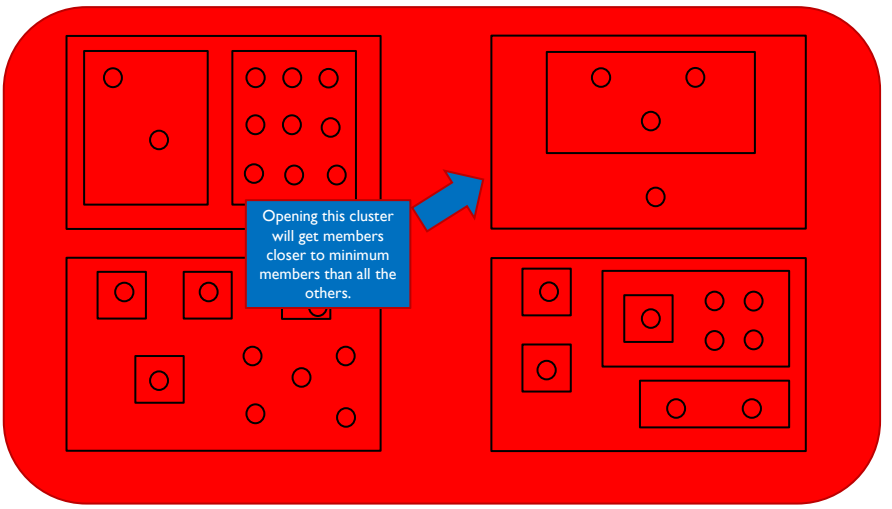
42

Example – Initial Clustering



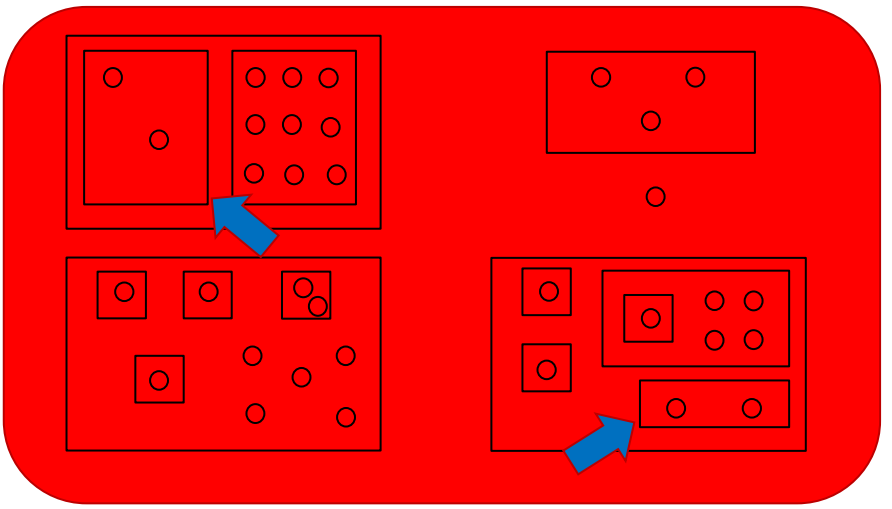
44

Example – Top-Down



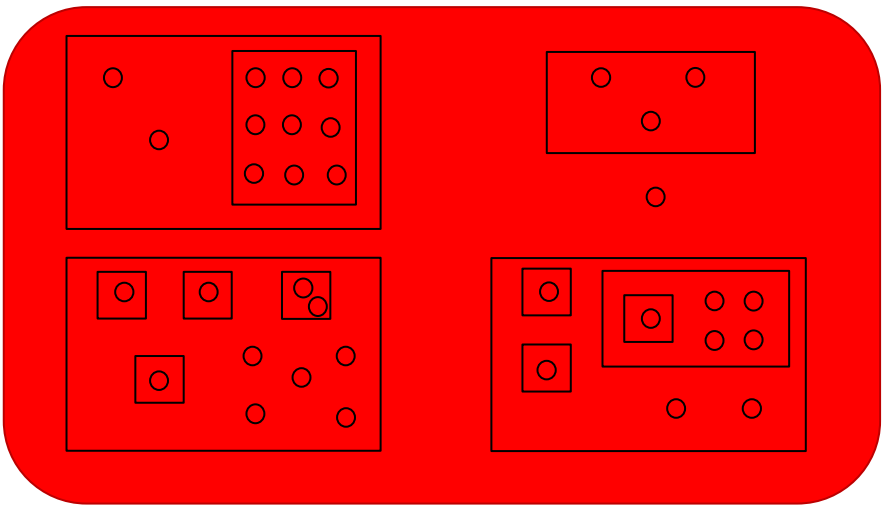
45

Example – Top-Down



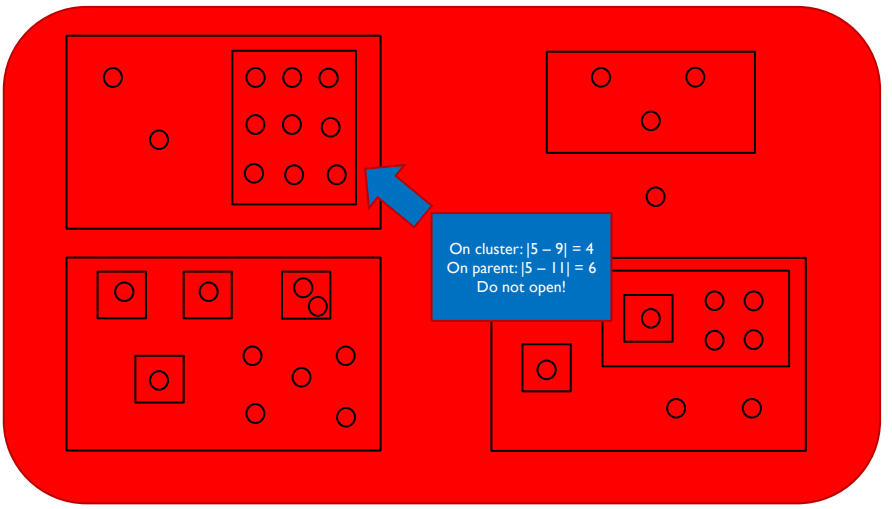
46

Example – Top-Down



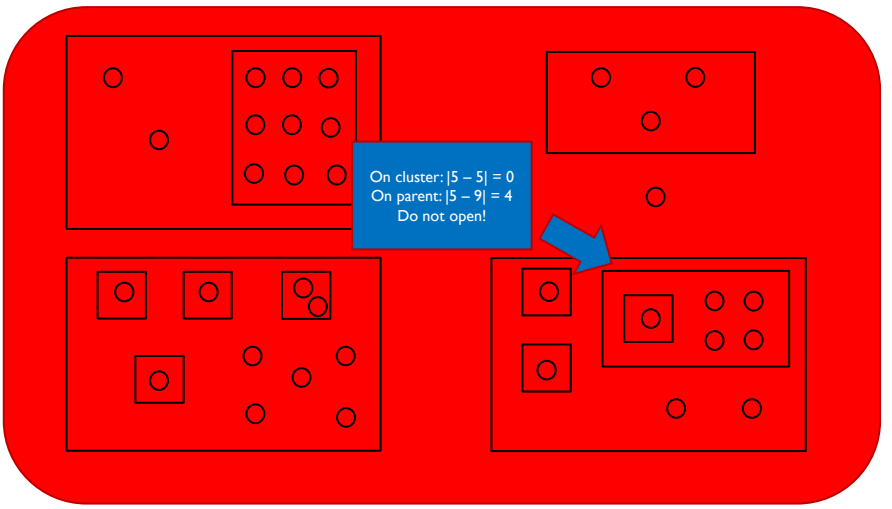
47

Example – Bottom-Up



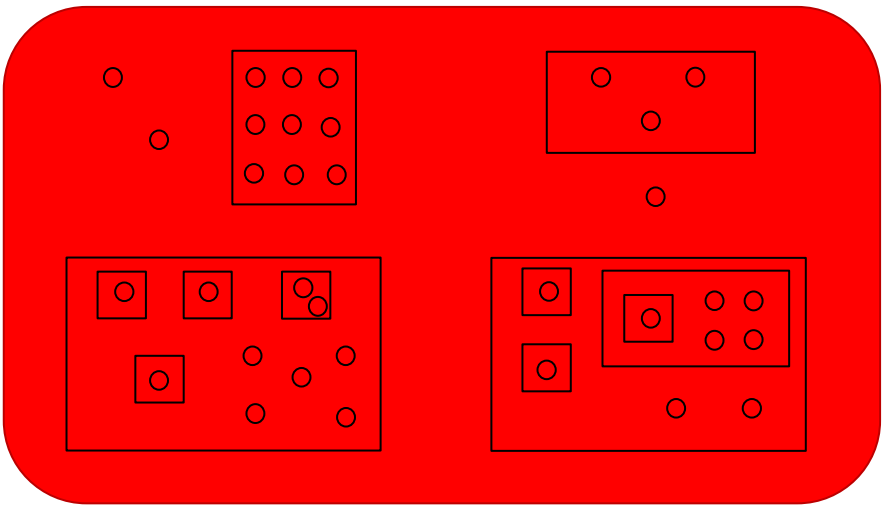
48

Example – Bottom-Up



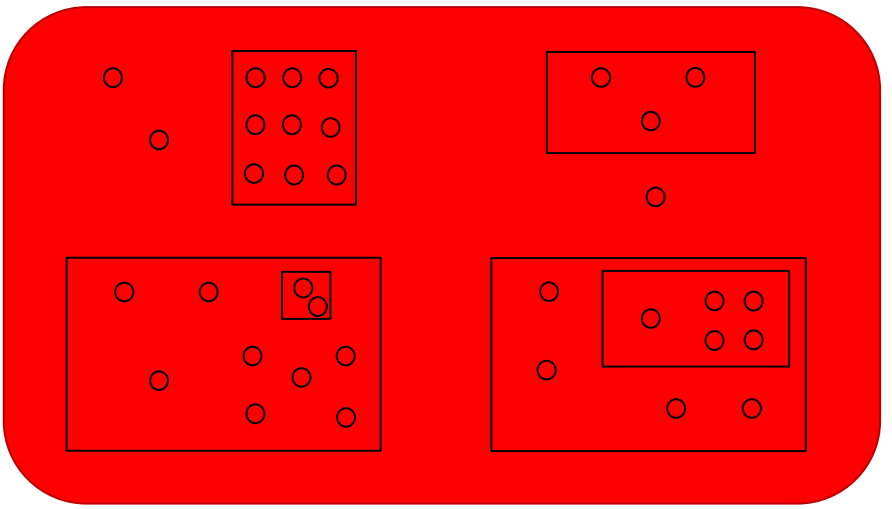
49

Example – Bottom-Up



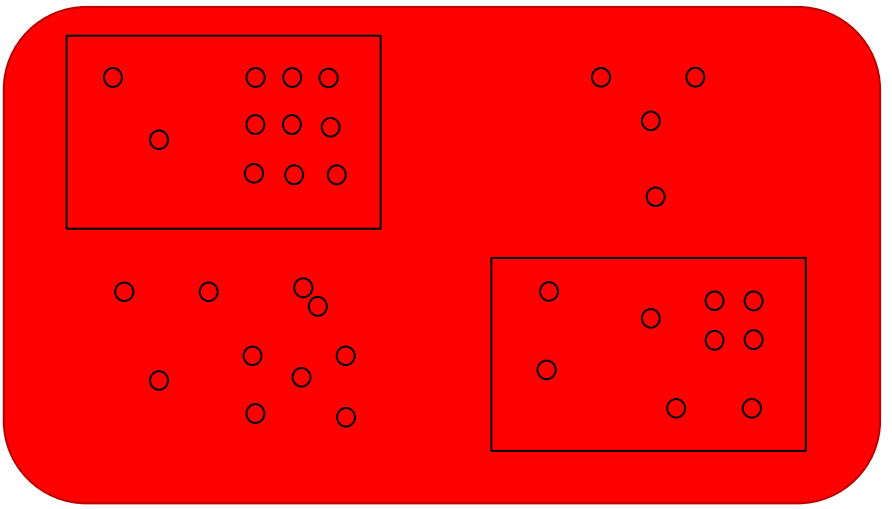
50

Example – Remove remaining elements



51

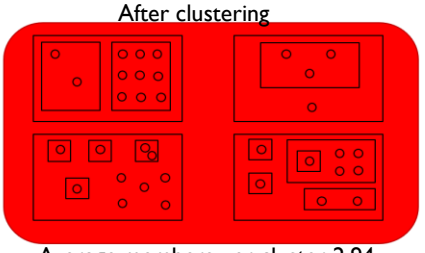
Example – Existing algorithm result



52

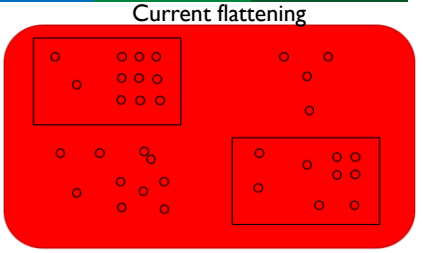
Comparison

After clustering



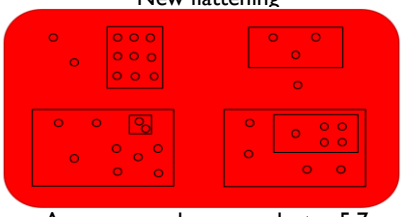
Average members per cluster 2.94

Current flattening



Average members per cluster 12

New flattening



Average members per cluster 5.7