1 Resampling

The places where resampling/averaging can play a part in the solution tree is illustrated in Fig. 1. The *Resampler* node changes the resolution of the result. The *ModRes* node changes the resolution of the request.

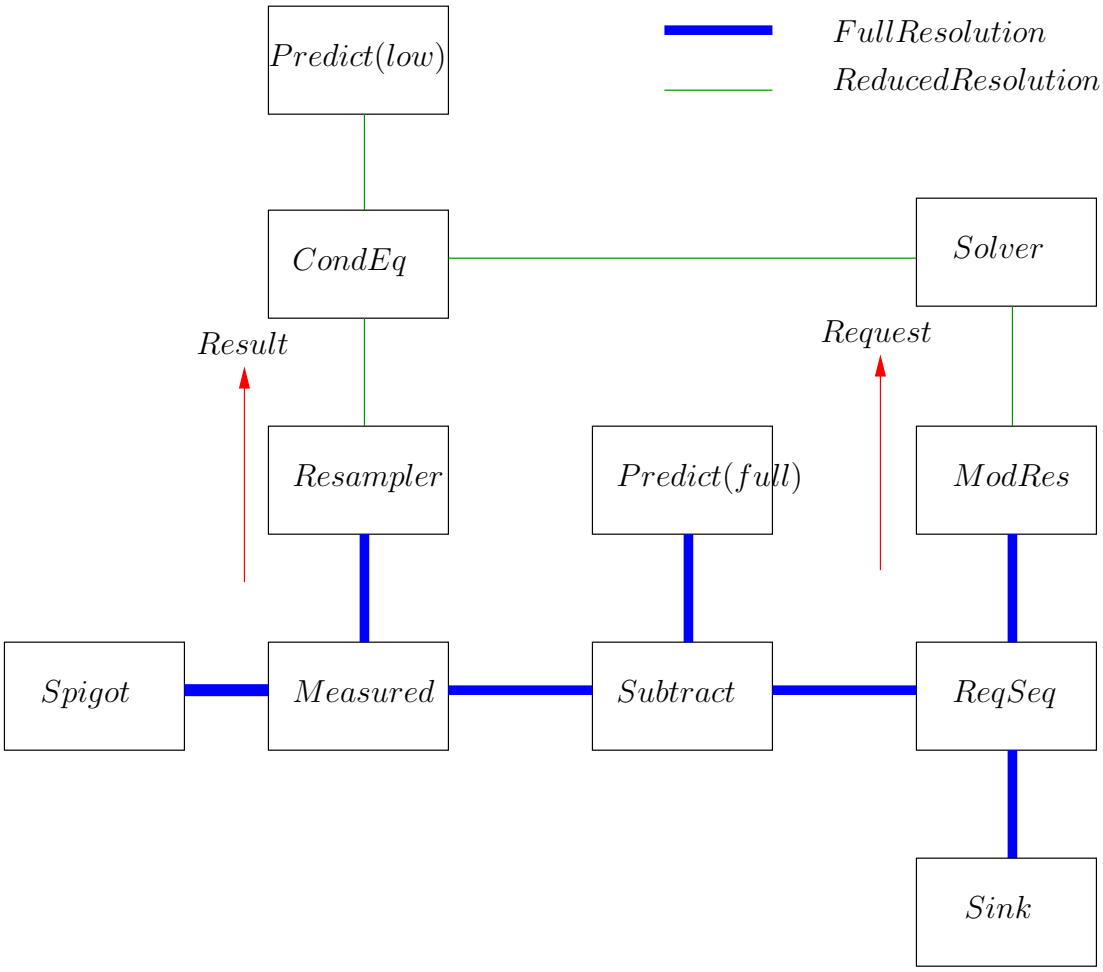

Figure 1: Solution tree with resampling

The high resolution request is generated by the MeqSink node. ModRes node will convert the high resolution request to a low resolution request. The Resampler gets a high resolution result from the Measuerd data. The resampler does a downsampling (Averaging) to convert the result to match the low resolution request.

In the following discussion we make the following assumptions.

- The cells are regular
- The resampled cells have the same domain as the original cells.

In Fig. 2 and 3 we see two common situations where resampling of the result takes place. In both cases, the resampling reduces the resolution (downsampling). Let us consider the original cells the have dimension $M \times N$ and the resampled cells to have the dimension $m \times n$.

In Fig. 2, m and n are factors of M and N respectively. Thus the original cells are exactly enclosed by the resampled cells. The resampling (Integration or Averaging) can be done in this case by considering the mn cells individually. On average, each new cell will include $\frac{M}{m} \times \frac{N}{n}$ old cells. Thus the cost of resampling (averaging) is $mn \times \frac{M}{m} \times \frac{N}{n}$ which is MN.

In contrast, the cells in Fig. 3, does not completely overlap. Hence, some cells will be resampled onto more than one cell. In this case, the straightforward way to resample (average) is by considering the MN original cells individually. Before doing that, we need to determine for each original cell the new cell which encloses it. Because the grid is regular, this can be done by binary search for each axis separately with cost $M \log m + N \log n$. After this is found, the cost of calculating the averages or contribution of each original cell is MN. Thus the total cost is $M \log m + N \log n + MN$.

1.1 Initial Parameters for Resampler

- `flag_mask` If the flag and ORed value of `flag_mask` is one, this cell is flagged, i.e. flag this cell if `flag_mask & flag` $\neq 0$.
- `flag_bit` Set the output flags if any to this value.

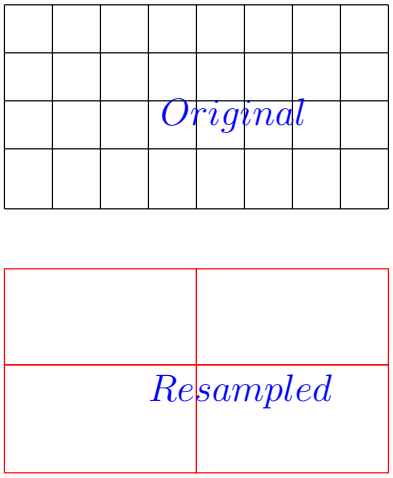

Figure 2: Exact resampling

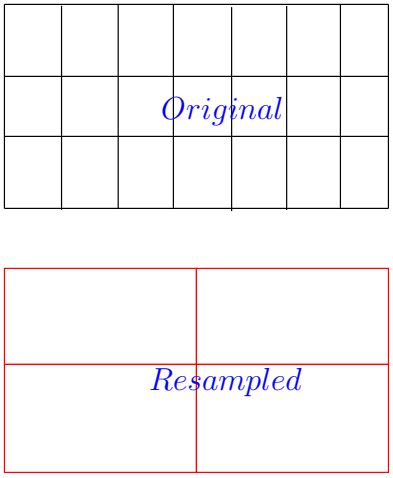

Figure 3: Inexact resampling

- `flag_density` If the ratio of total flagged cells over the total cells included in a new cell is greater than this value, flag the output.

1.2 Initial Parameters for ModRes

- `num_cells` This is an array of integers for the number of required cells in each dimension.

2 Implementation

If the cells are regular and the domain is fixed, there is a generalized implementation for downsampling, upsampling and mixed sampling. If we consider only one axis, this can be summerized as in Fig. 4 and Fig. 5. The key to this implementation is the construction of the bipartite graph. The edge weights are calculated as

$$edge\ weight = \frac{intersecting\ length}{original\ segment\ length}.$$

Once the graphs are created for each axis, the actual resampling is done as follows.

```
for i in  output nodes of axis 1,
  for k in incident edges to node i
   w_{i,k}=weight of the k-th inciden edge to node i
   for j in   output nodes of axis 2
    for l in incident edges to node l
     w_{j,l}=weight of the l-th incident edge to node j
        output value for cell (i,j) += w_{i,k} * w_{j,l}
```

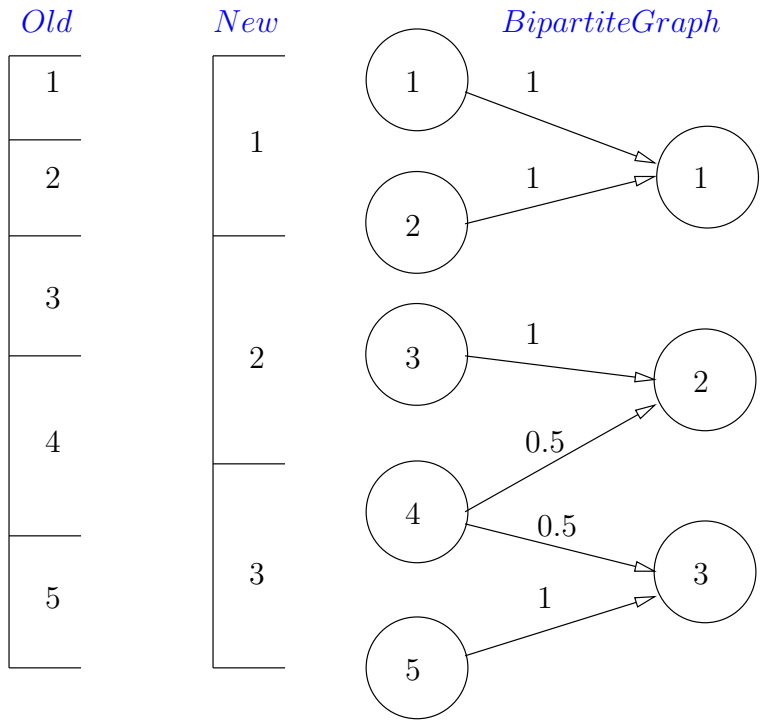

Figure 4: Downsampling

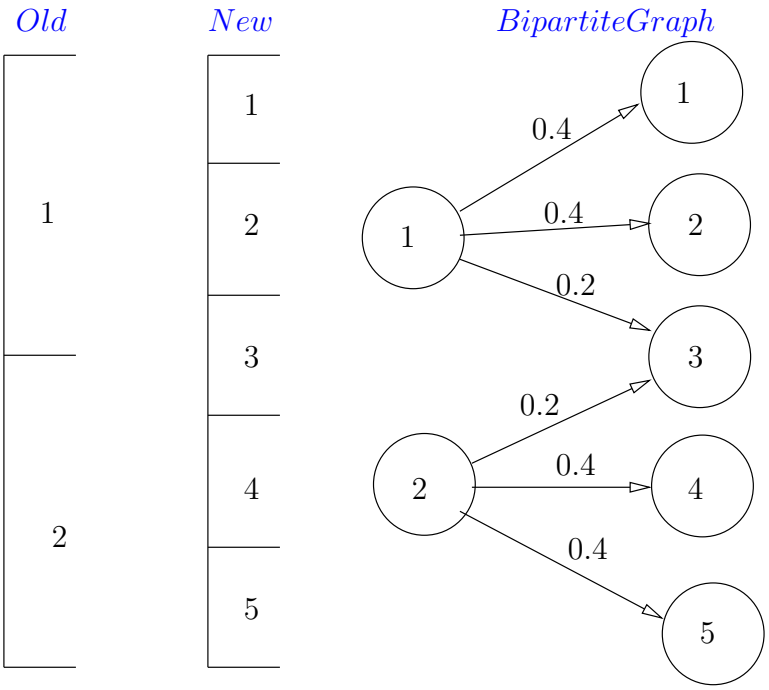

Figure 5: Upsampling