Paolo Calao – Gruppo 22

Low-Power Contest 17/18

Synthesis and Optimization of Digital Systems

Different approaches, even quite complex, have been taken in consideration and tried, but this simple algorithm is the one that results in best performances. Its parameters have been chosen after several attempts and tests.

Algo's behavior

- *Loop while (leakage_saved < required_savings AND NOT all cells are already swapped to hvt).*
  - → *Sort lvt_cell_list such that the first cells of the list are the best to be swapped to hvt. The sort is based on certain* ***parameters (I)*** *of the cells.*
  - → *Choose a* ***number of cells (II)*** *to be swapped proportional to the amount of leakage to be saved.*
  - → *Swap the cells and remove them from the lvt_cell_list.*

I. ***Parameters*** extracted from the cells to calculate their index of priority are: slack, leakage and delay.
These values are firstly normalized between 0 and 1. Secondly they are multiplied by different constants, then their sum is returned.
The best constants found are the most obvious:
k_slack = 1, k_delay = -1, k_leakage = 1.
This remarks that the best compromise is in the half.

II. The ***number of cells*** to be swapped during the current iteration is quite important. On one hand if too many cells are swapped it is probable that too much leakage with respect to the constraints would be saved, consequently the slack would be worse than the necessary. On the other hand if the number of cells swapped per iteration is too small then the time of computation required by the script would be unnecessarily big.
The best amount of swapped cells per iteration found is the following:
num_cells = num_lvt_cells * 0.65 * (required_savings - current_savings).