# Force Directed Scheduling

By : Nirojan Navaratnarajah

# Motivation

# Force Directed Scheduling

- The algorithm computes a final schedule by minimizing the total energy of the system, subject to resource constraints and task dependencies.

- The objective is to decrease the number of processors needed by reducing the level of simultaneous execution of operations assigned to them, while maintaining the same overall execution duration[1]

# ASAP & ALAP Scheduling

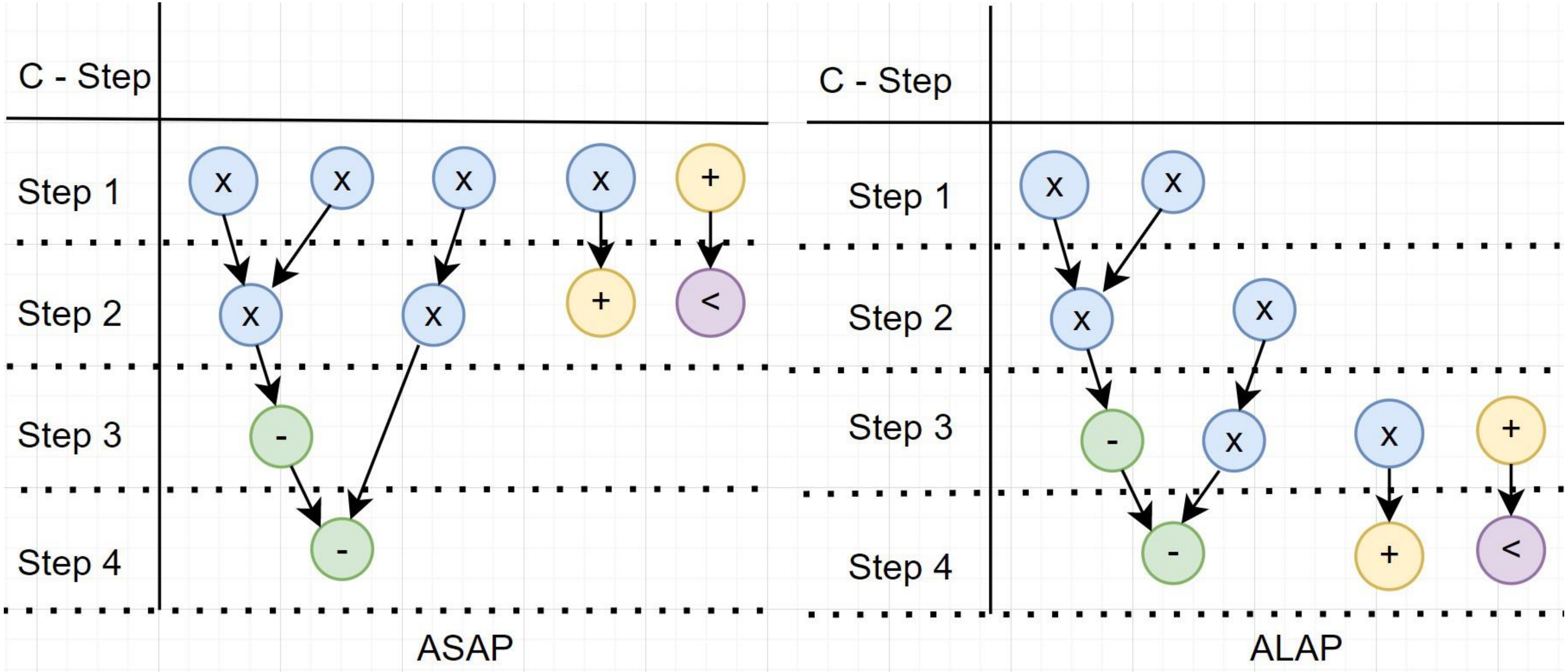
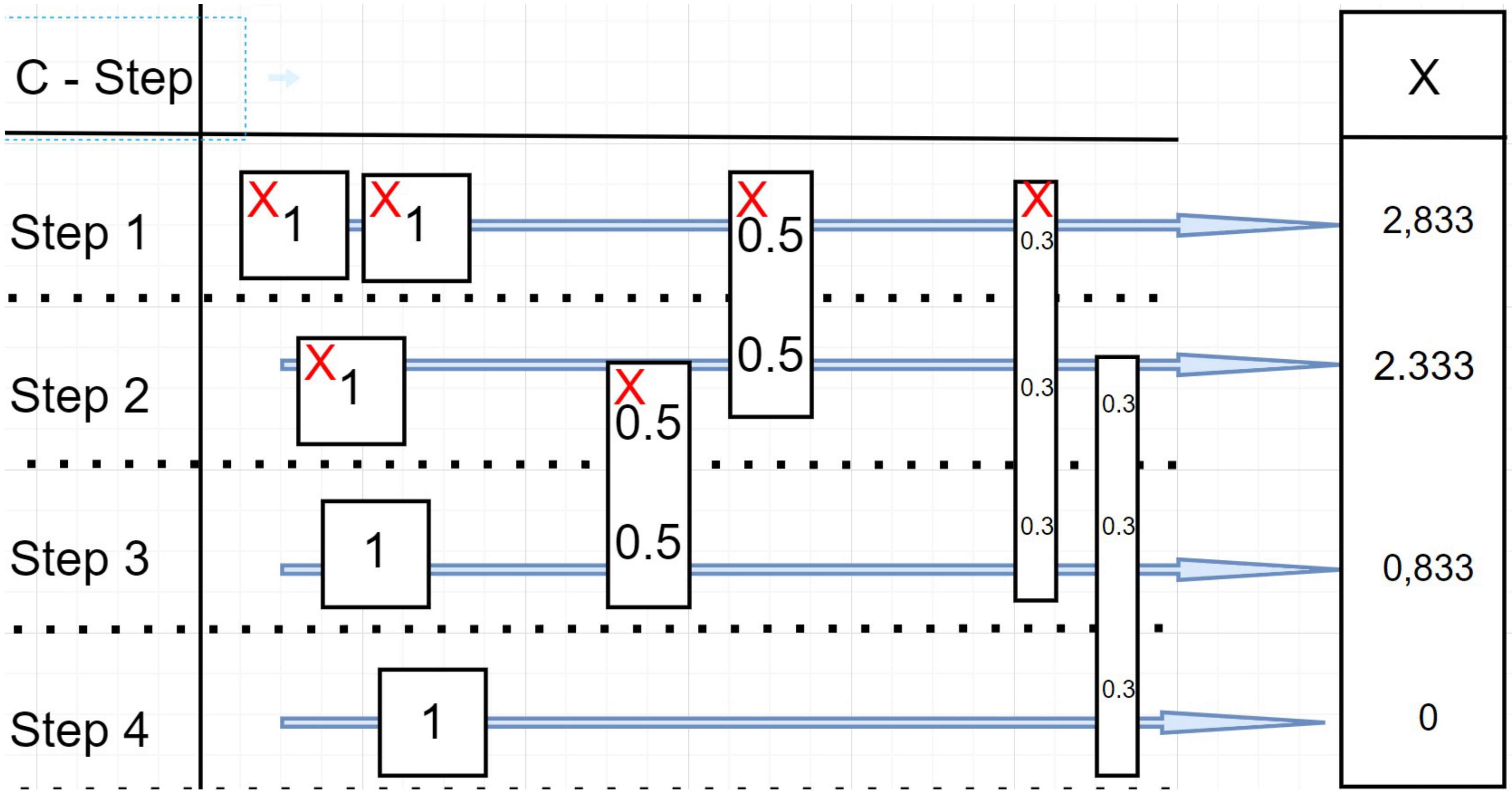



FIG 1 . based on [4]

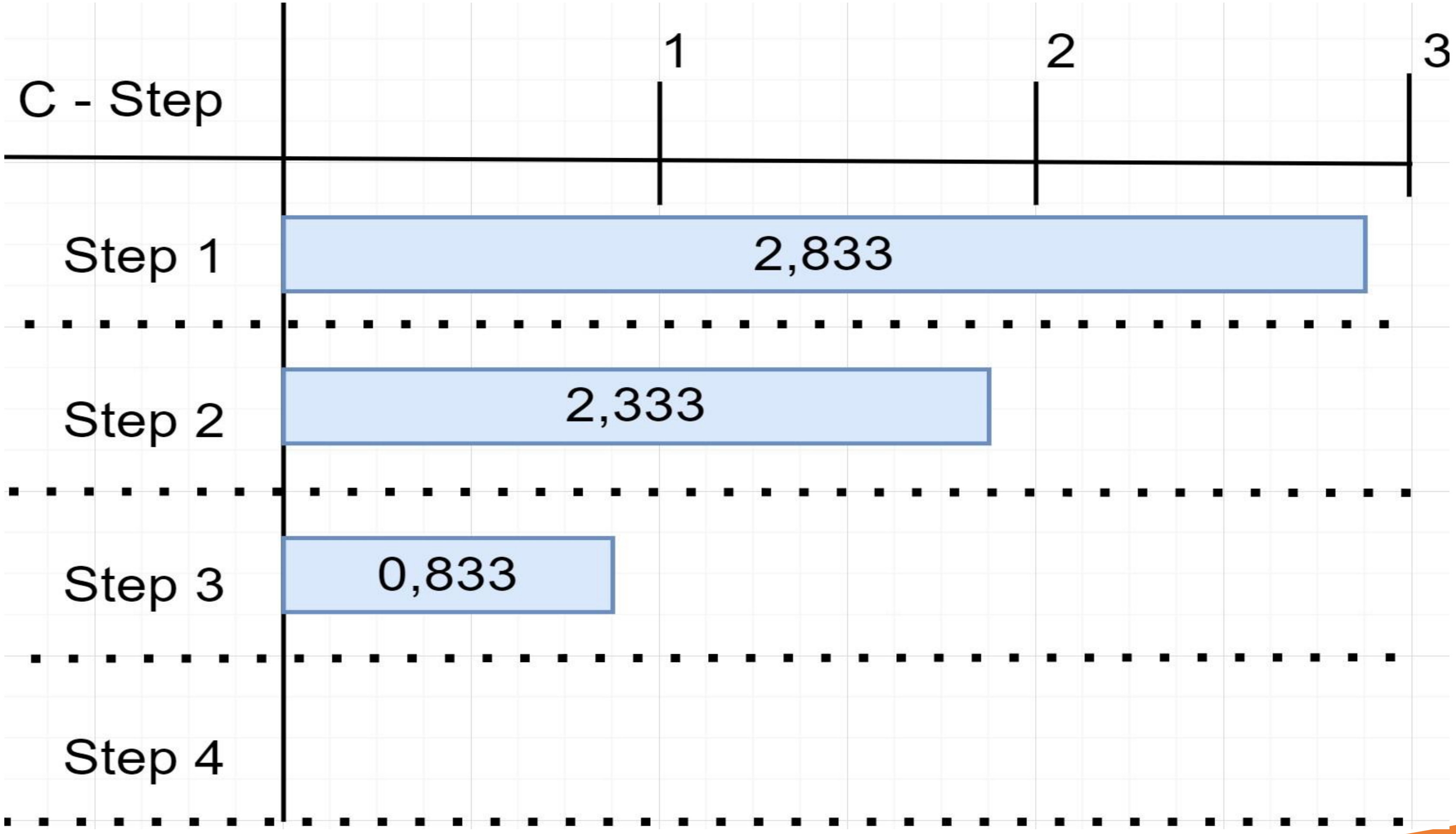
# Diagram Flow Graph



# Steps for determining the Scheduling

1. Determine time frames for operations.
2. Update the distribution graphs, considering any conditional dependencies.
3. Calculate self-forces for each operation.
4. Calculate forces from predecessor/successor operations.
5. Schedule the operation and its corresponding c-step pair that yields the best outcome.

# Initial Distribution Graph



# Hooke’s Law

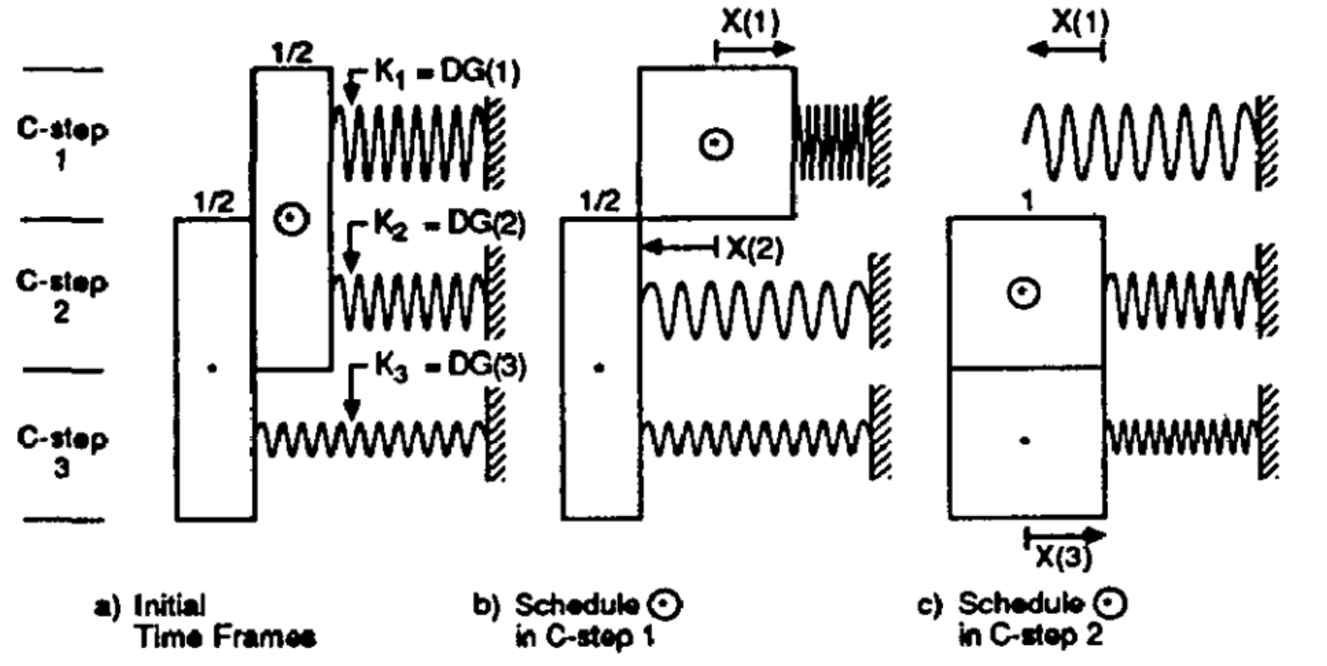



Figure 6. Time frame modifications for force calculations.

$$F = K \cdot x$$

[1]

# Force Calculation for Mobile Operations

- In C-Step 1 :
  Force = 2.833*(1-0.5) + 2.333*(0-0.5) = + 0.25

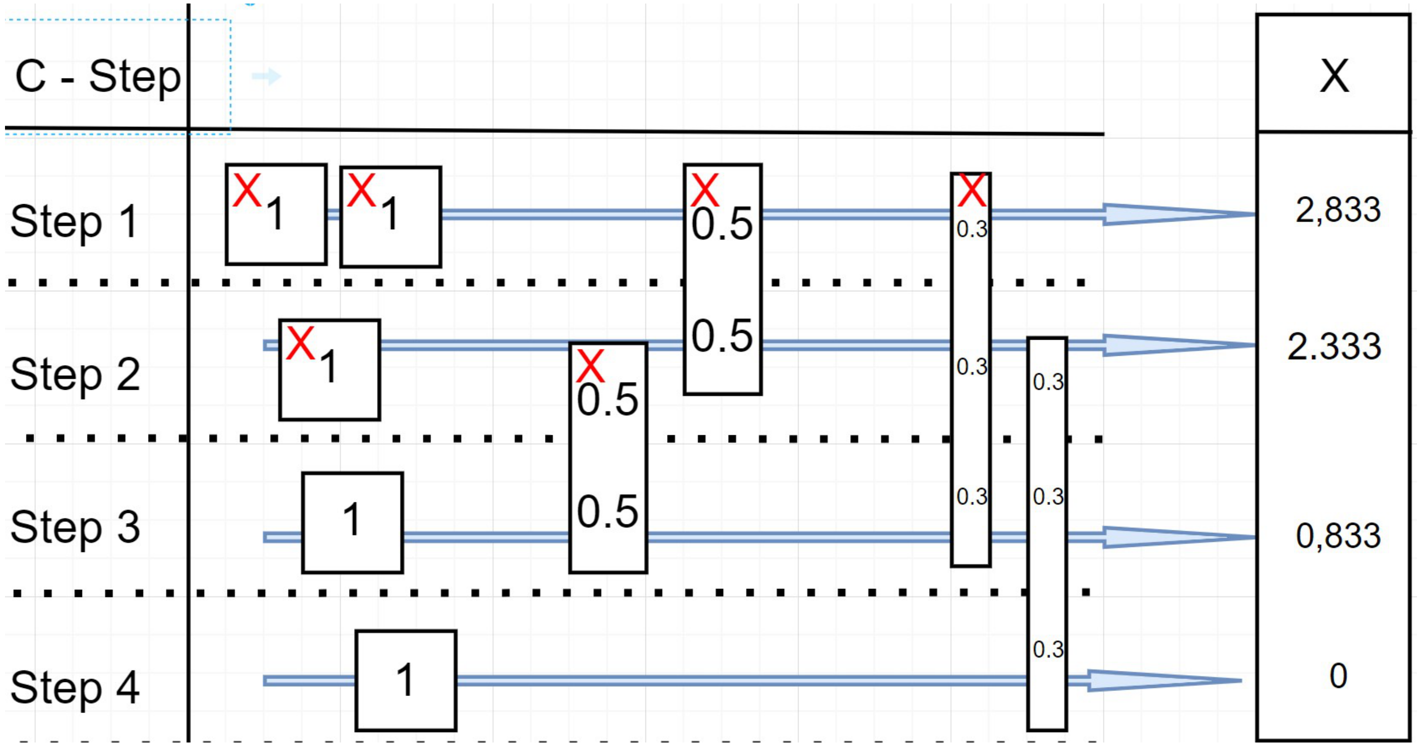
- In C-Step 2 : F = 2.833*(0-0.5) + 2.333*(1-0.5)= - 0.25

For the first node with mobility in C step 2 :

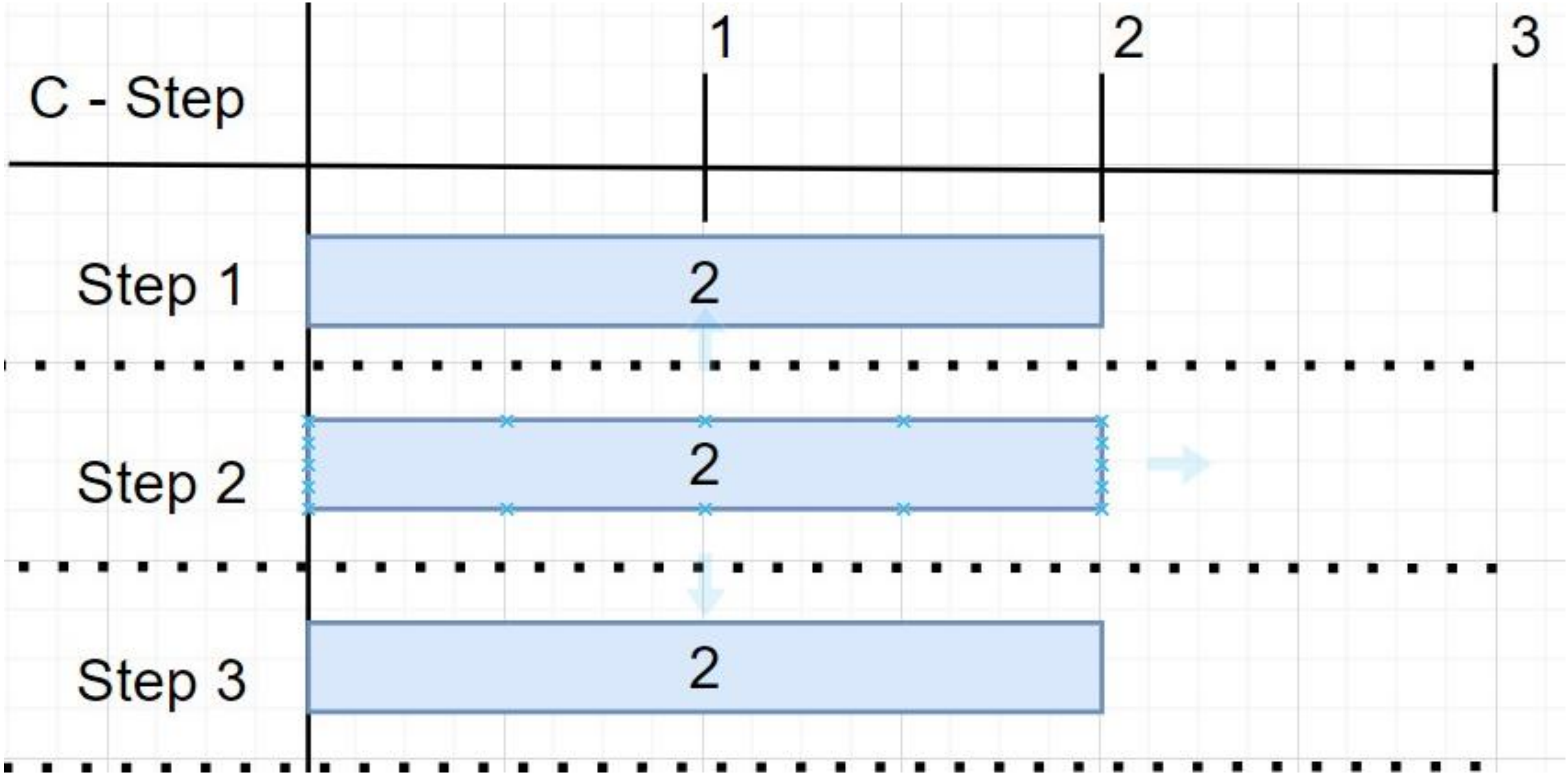
- In C-Step 2 : 2.333*(1-0.5) + 0.833*(0-0.5) = + 0.75
- In C-Step 3 : 2.333*(0-0.5) + 0.833*(1-0.5) = - 0.75

For last node :

- In C-Step 1 :
  F = 2.833*(1-0.33) + 2.333*(0-0.33) + 0.833*(0-0.33)
  = + 0.853
- In C- Step 2 :
  F =2.833*(0-0.33)+2.33*(1-0.33)+0.833*(0-0.33)=0.351



# Final distribution Graph after using FDS



# Advantages of FDS

- Resource optimization to minimize idle time and improve resource utilization.
- Identification of parallel task execution opportunities to increase efficiency.
- Real-time adaptation to respond to new tasks or constraints.
- Visual representation for better understanding and analysis.
- Conflict resolution to handle overlapping requirements.
- Flexibility to accommodate changes and uncertainties.
- Scalability for handling large and complex projects.[8]

# Summary

1.FDS is a powerful approach in hardware-software codesign that optimizes task scheduling by simulating forces to guide task allocation and minimize conflicts.

2.It offers flexibility to accommodate changes and uncertainties, allowing for dynamic adjustments during the design process.

3.FDS optimizes resource utilization by minimizing idle time and identifying opportunities for parallel task execution.

4.Its conflict resolution capabilities enable efficient handling of overlapping requirements and dependencies.

5.With visual representation and real-time adaptation, force-directed scheduling enhances understanding, analysis, and the ability to respond to new tasks or constraints.

# References and Further Reading

[1] P. G. Pauline and J. P. Knight, "Force-Directed Scheduling in Automatic Data Path Synthesis," 24th ACM/IEEE Design [2] R. G. Johnson, "Heterogeneous parallel processing," Communications of the ACM, vol. 26, pp. 832-844, 1983.
[3] F. Brandenburg, "Force Synthesis and optimization of digital circuits."
[4] E. T. Lee and J. B. Schrage, "An algorithm for scheduling independent tasks," Operations Research, vol. 22, pp. 458-475, 1974.
[5] C. K. Lee, S. C. Tan, F. F. Wu, S. Y. R. Hui and B. Chaudhuri, "Use of Hooke’s law for stabilizing future smart grid — The electric spring concept," 2013 IEEE Energy Conversion Congress and Exposition, Denver, CO, USA, 2013, pp. 5253-5257, doi: 10.1109/ECCE.2013.6647412.
[6] Scheduling Algorithms for High-Level Synthesis Zoltan Baruch Computer Science Department, Technical University of Cluj-Napoca, URL - http://users.utcluj.ro/baruch/papers/Scheduling-Algorithms.pdf
[7] M. Fons, F. Fons and E. Canto, "Hardware-Software Codesign of a Fingerprint Alignment Processor," 2007 14th International Conference on Mixed Design of Integrated Circuits and Systems, Ciechocinek, Poland, 2007, pp. 661-666, doi: 10.1109/MIXDES.2007.4286246.
[8] De Micheli, G. Synthesis and Optimization of Digital Circuits.
[9] W. F. J. Verhaegh, E. H. L. Aarts, J. H. M. Korst and P. E. R. Lippens, "Improved force-directed scheduling," Proceedings of the European Conference on Design Automation, Amsterdam, Netherlands, 1991, pp. 430-435, doi: 10.1109/EDAC.1991.206441.
[10] C. Xia and G. Yang, "Implementation of DMA driver development under Linux," 2021 IEEE 2nd International Conference on Information Technology, Big Data and Artificial Intelligence (ICIBA), Chongqing, China, 2021, pp. 721-724, doi: 10.1109/ICIBA52610.2021.9688147.