Cobot-Human Assembly Line Balancing Problem

Tevhide Altekin,
Yossi BukchinYossi Bukchin ,
Tel-Aviv University,
`

Abstract

In this paper, we consider a manual assembly line consisting of multiple stations, each accommodating a worker. Collaborative robots (Cobots) are assigned to stations in order to assist the workers by performing part of the workload. The role of the cobot is to replace the worker in some of the tasks, such as positioning parts, pick& place, or for example, holding an item while the worker performs some task on it. The assembly operation is captured by a precedence diagram, consisting of the tasks needed to be done to complete the product, with precedence relations among them. Each task can be classified as follows: (1) worker task; (2) cobot task; (3) either task (can be done by either the cobot or the worker); (4) concurrent task (needs both the worker and the cobot). The objective is to allocate the workload among the workers and cobots, given the line configuration, to minimize the cycle time (or maximize the throughput rate).

The following collaboration types are defined: (1) Independent collaboration: workers and cobots work on separate tasks. Each task is solely done by either the human or the cobot; (2) Concurrent collaboration: a worker and a cobot perform together the same task (e.g., the cobot holds the part while the human makes some operation); (3) Sequential collaboration: two tasks, one done by the worker (cobot) and the other by the cobot (worker) have to be done *continuously* one after the other.

Given the configuration of workers and cobots, the problem is to assign tasks to workers and cobots while minimizing the cycle time. Since the problem is solved for a given configuration, we define dominance rules among configurations and show that only non-dominated configurations should be considered. We prove some properties of such configurations. Next, a MILP formulation is developed to solve moderate problem sizes. To improve the performance of the formulation, we develop some bounds on the cycle time. Then we demonstrate the performance of this formulation to solve problems of up to 20 tasks to optimality and up to 50 tasks with a relatively small optimality gap. For solving large-scale problems, we develop an Adaptive Large Neighborhood Search (ALNS) heuristic, which repeatedly improves an initial solution by destroying and repairing the solution. The former is done by removing tasks from the solution, while the latter is done by solving a modified MILP formulation. The performance of the heuristic is examined in wide experimentation.