MultiObjective Decision Analysis in Job Scheduling

Zeynep Altıner

Ayşe Beyza Çanakçı

Simge Çınar

Instructor: Assoc. Prof. Özlem Karsu

May 30, 2023

Agenda

Introduction

- Scheduling problem with parallel machines and splitting jobs

- NP-hard

- Multiobjective VPL algorithm is used in the paper

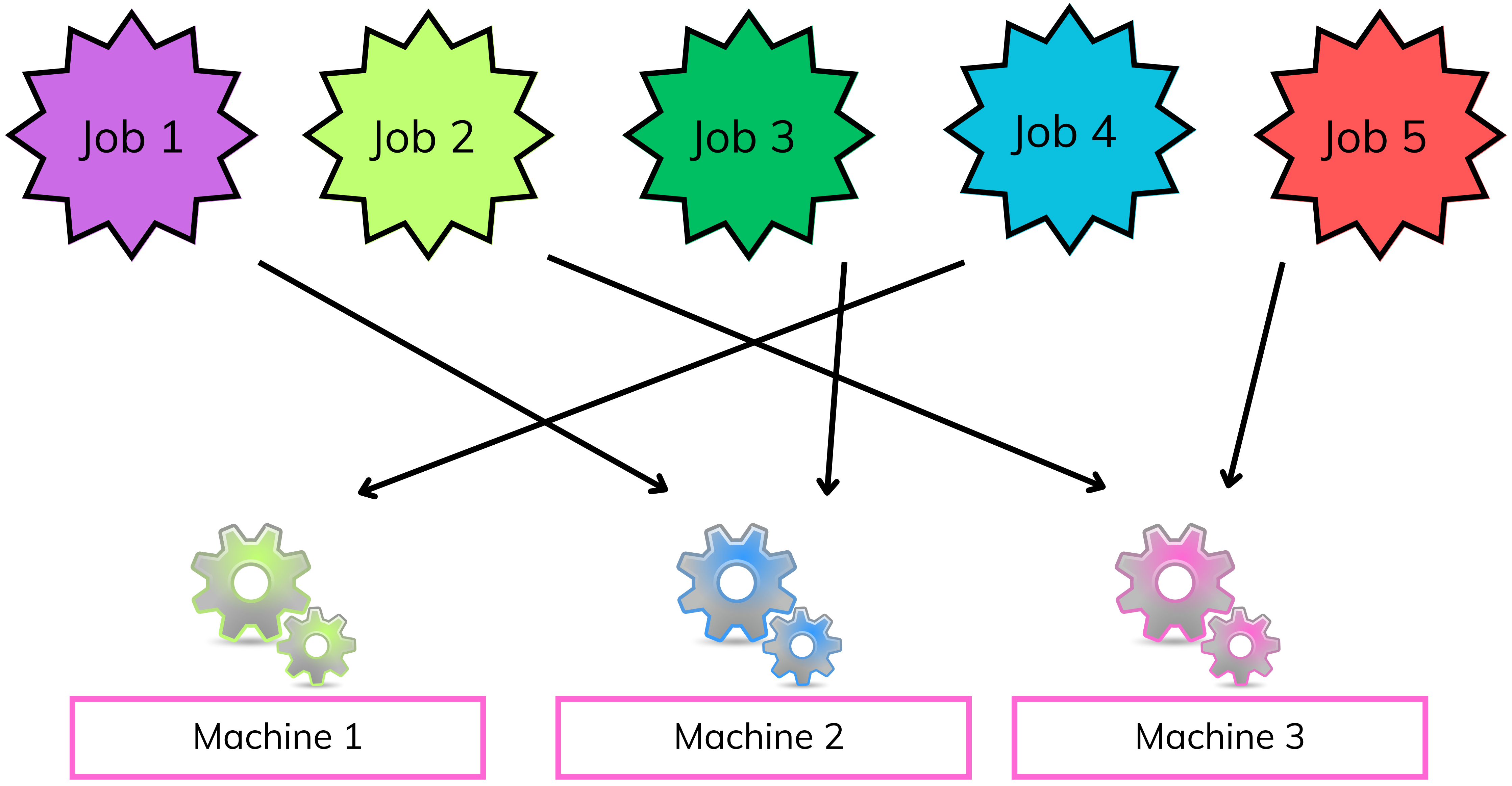
A multi objective volleyball premier league algorithm for green scheduling identical parallel machines with splitting jobs

Khodakaram Salimifard[1] · Jingpeng Li[2] · Davood Mohammadi[1] · Reza Moghdani[1]

Accepted: 16 October 2020 / Published online: 8 December 2020

Abstract
Parallel machine scheduling is one of the most common studied problems in recent years, however, this classic optimization problem has to achieve two conflicting objectives, i.e. minimizing the total tardiness and minimizing the total wastes, if the scheduling is done in the context of plastic injection industry where jobs are splitting and molds are important constraints. This paper proposes a mathematical model for scheduling parallel machines with splitting jobs and resource constraints. Two minimization objectives - the total tardiness and the number of waste - are considered, simultaneously. The obtained model is a bi-objective integer linear programming model that is shown to be of NP-hard class optimization problems. In this paper, a novel Multi-Objective Volleyball Premier League (MOVPL) algorithm is presented for solving the aforementioned problem. This algorithm uses the crowding distance concept used in NSGA-II as an extension of the Volleyball Premier League (VPL) that we recently introduced. Furthermore, the results are compared with six multi-objective metaheuristic algorithms of MOPSO, NSGA-II, MOGWO, MOALO, MOEA/D, and SPEA2. Using five standard metrics and ten test problems, the performance of the Pareto-based algorithms was investigated. The results demonstrate that in general, the proposed algorithm has supremacy than the other four algorithms.

Problem Definiton

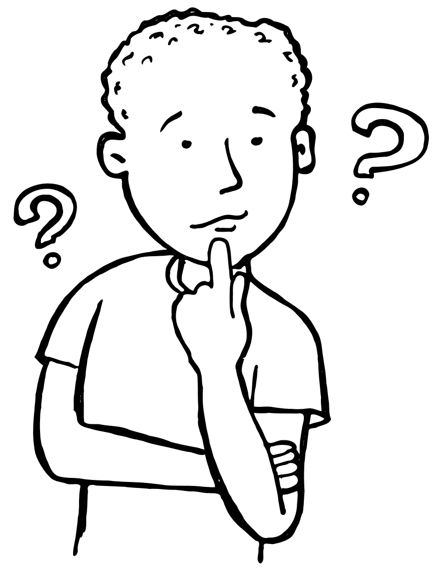


GOAL!

Minimize tardiness

Minimize number of defective outputs

How to assign jobs to machines ?



Modified Mathematical Model

Parameters

Parameters:

MO_i: Total number of machines that can be used to perform job i

P_i : Processing time of job i

Q_i : The amount of jobs

$S_{i,j}$: Setup time in case job j comes after job i

d_i : Due data of job i

Wij: Sequence dependent setup defective output of job j if it comes after job i.

a_k: Amount of defective output of machine k.

Decision Variables

Decision Variables:

$$x_{0jk} = \begin{cases} 1, & \text{if job } j \text{ is the first job on machine } k \\ 0, & \text{otherwise} \end{cases}$$

$$y_{ik} = \begin{cases} 1, & \text{if job } i \text{ is processed on machine } k \\ 0, & \text{otherwise} \end{cases}$$

$$x_{i0k} = \begin{cases} 1, & \text{if job } i \text{ is the last job to be processed on machine } k \\ 0, & \text{otherwise} \end{cases}$$

$$x_{ijk} = \begin{cases} 1, & \text{if job } j \text{ comes after job } i \text{ on machine } k \\ 0, & \text{otherwise} \end{cases}$$

Z_i: Tardiness for job i

q_{ik}: Amount of job i on machine k

C_{ik}: Completion time of job i on machine k

Assumptions

When an amount of job is assigned to a machine, it will be completely processed in that machine

There is no setup time for initial jobs

Number of defective outputs of each machine is limited by DM

All machines are identical

The same deadline applies for job i

All machines must be used for at least for one job

Modified Mathematical Model

$$\text{Min } z_1 = \sum_{i \in I} Z_i$$

$$\text{Min } z_2 = \sum_{i \in I, j \in J, k \in K} x_{ijk} \cdot Wij$$

subject to

$$\sum_{k \in K} q_{ik} = Q_i, \quad i \in I \tag{1}$$

$$y_{ik} \geq \frac{q_{ik}}{Q_i}, \quad i \in I, \ k \in K \tag{2}$$

$$y_{ik} \leq q_{ik}, \quad i \in I, \ k \in K \tag{3}$$

$$x_{0jk} + \sum_{i \in I} x_{ijk} = y_{jk}, \quad j \in J, \ k \in K \tag{4}$$

$$x_{j0k} + \sum_{j \in J} x_{ijk} = y_{ik}, \quad i \in I, \ k \in K \tag{5}$$

$$x_{0jk} \cdot (q_{ik} \cdot P_i) \leq C_{ik} \cdot y_{ik}, \quad i \in I, \ k \in K \tag{6}$$

$$x_{ijk} \cdot (S_{i,j} + C_{ik}) + (q_{jk} \cdot P_j) \leq C_{ik} \cdot y_{jk}, \quad i \in I, \ j \in J, \ k \in K \tag{7}$$

$$C_{ik} - d_i \leq Z_i \quad i \in I, \ k \in K \tag{8}$$

$$x(iik) = 0 \quad i \in I, \ k \in K \tag{9}$$

$$\sum_{i \in I, j \in J} x_{ijk} \cdot Wij \leq a_k \quad k \in K \tag{10}$$

$$\sum_{k \in K} y_{ik} \leq MO_i \quad i \in I \tag{11}$$

$$\sum_{j \in J} x_{0jk} = 1 \quad k \in K \tag{12}$$

$$\sum_{j \in J} x_{j0k} = 1 \quad k \in K \tag{13}$$

$$\sum_{i \in I} y_{ik} \geq 1 \quad k \in K \tag{14}$$

$$C_{ik} \leq M \cdot y_{ik} \quad i \in I, \ k \in K \tag{15}$$

$$C_{ik} \geq y_{ik} \quad i \in I, \ k \in K \tag{16}$$

$$x_{0jk} \in \{0,1\}, x_{j0k} \in \{0,1\} \quad j \in J, \ k \in K \tag{17}$$

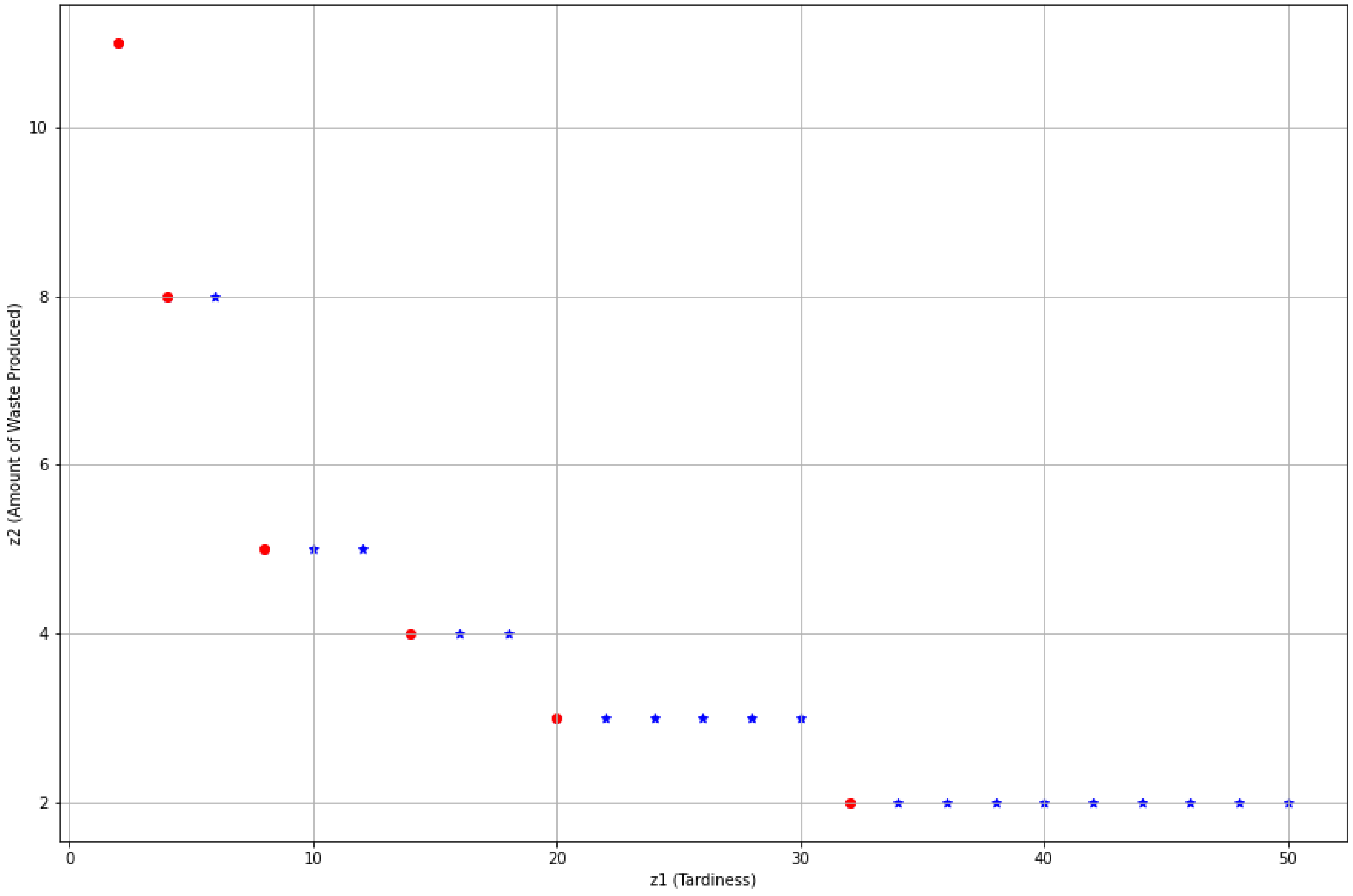
$$x_{ijk} \in \{0,1\} \quad i \in I, \ j \in J, \ k \in K, y_{jk} \in \{0,1\} \quad i \in I, \ k \in K \tag{18}$$

$$q_i \geq 0 \quad i \in I, \ k \in K, Z_i \geq 0 \quad i \in I, C_{ik} \geq 0 \quad i \in I, \ k \in K \tag{19}$$

$$\text{all dv's are integer} \tag{20}$$

Why ε-Constraint Algorithm ?

Pre-analysis

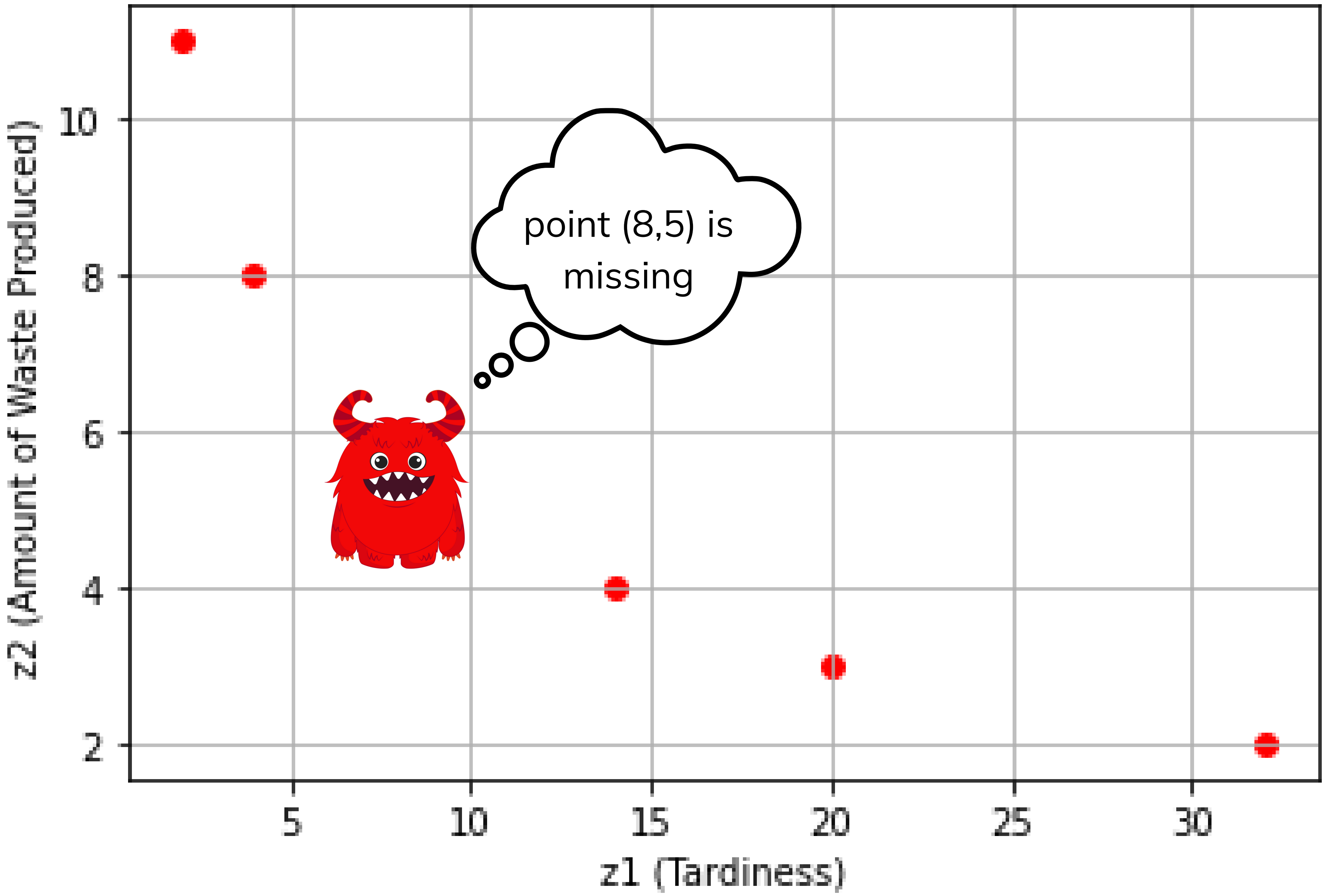


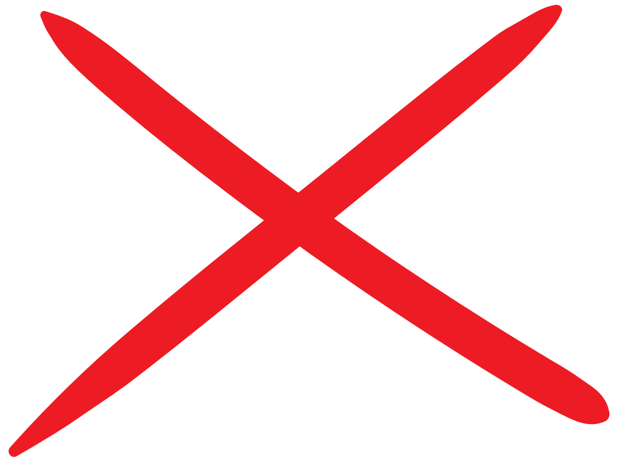
epsilon = 50
step size = 1
iterations = 50

epsilon =32
step size = 1
iterations = 32

epsilon = 32
step size = 2
iterations = 16

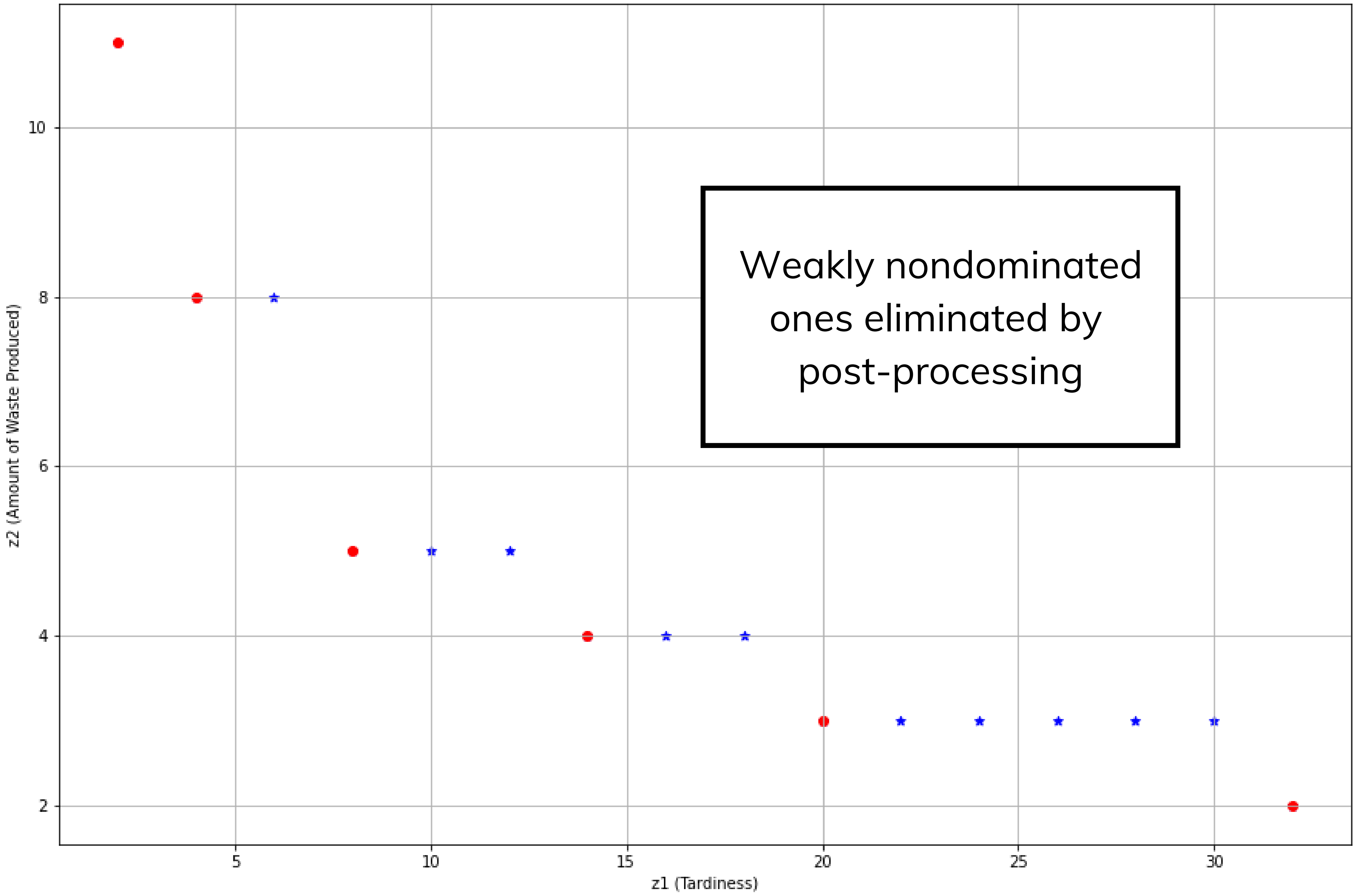
Pre-analysis





epsilon = 32
step size = 3

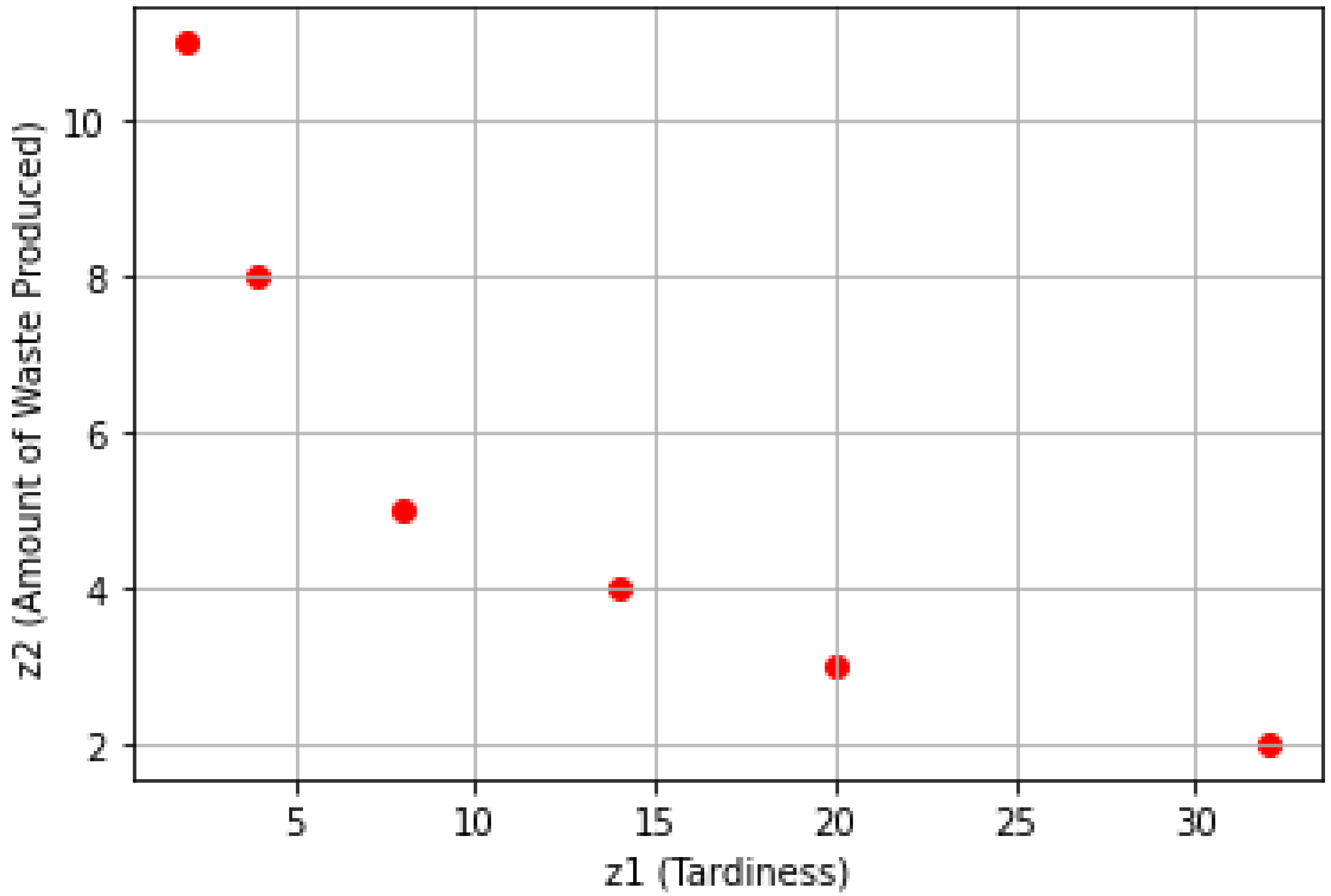
Findings for the ε-Constraint Algorithm



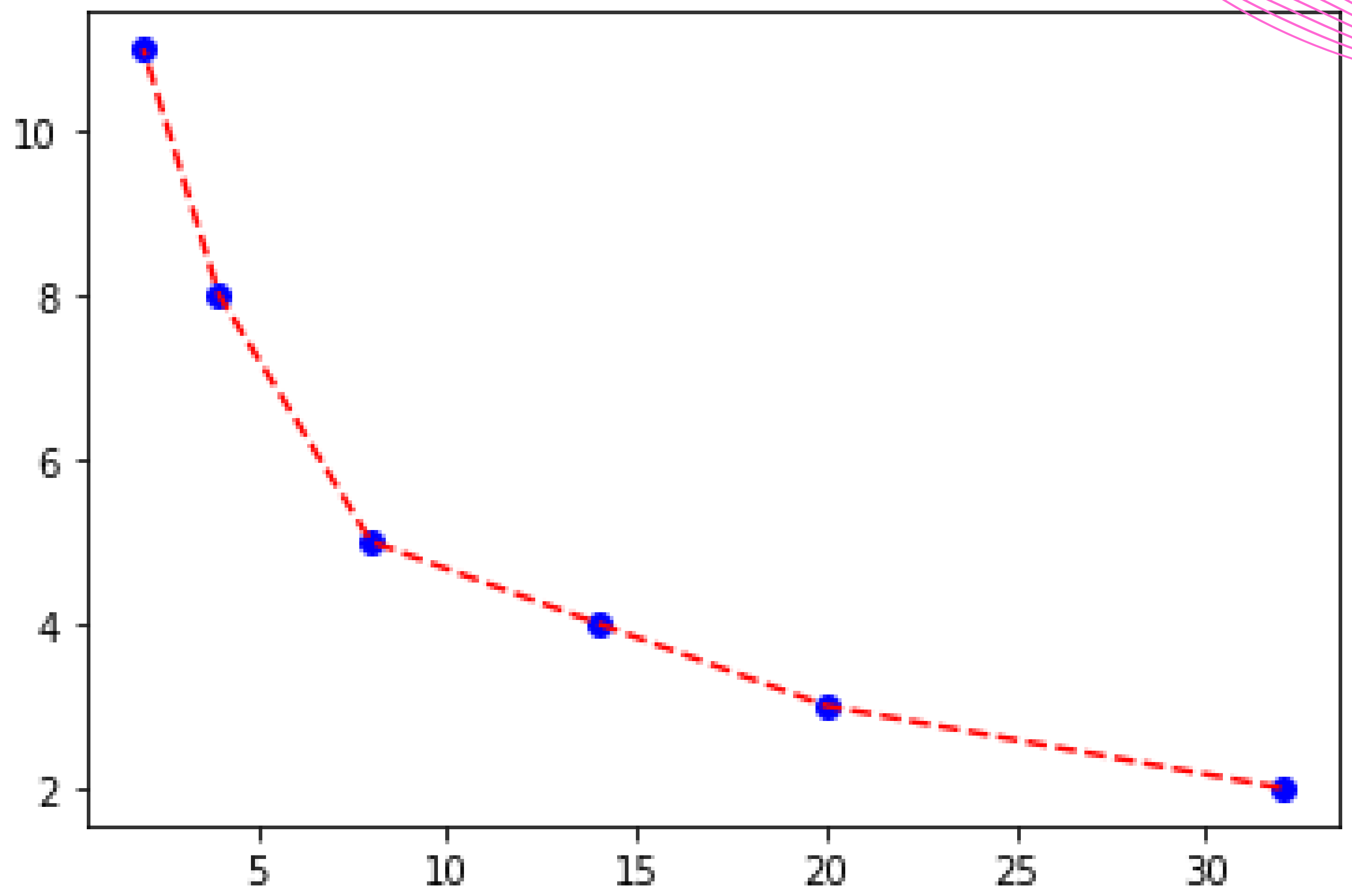
Weakly nondominated points

epsilon = 32
step size = 2
16 iterations

Findings for the ε-Constraint Algorithm

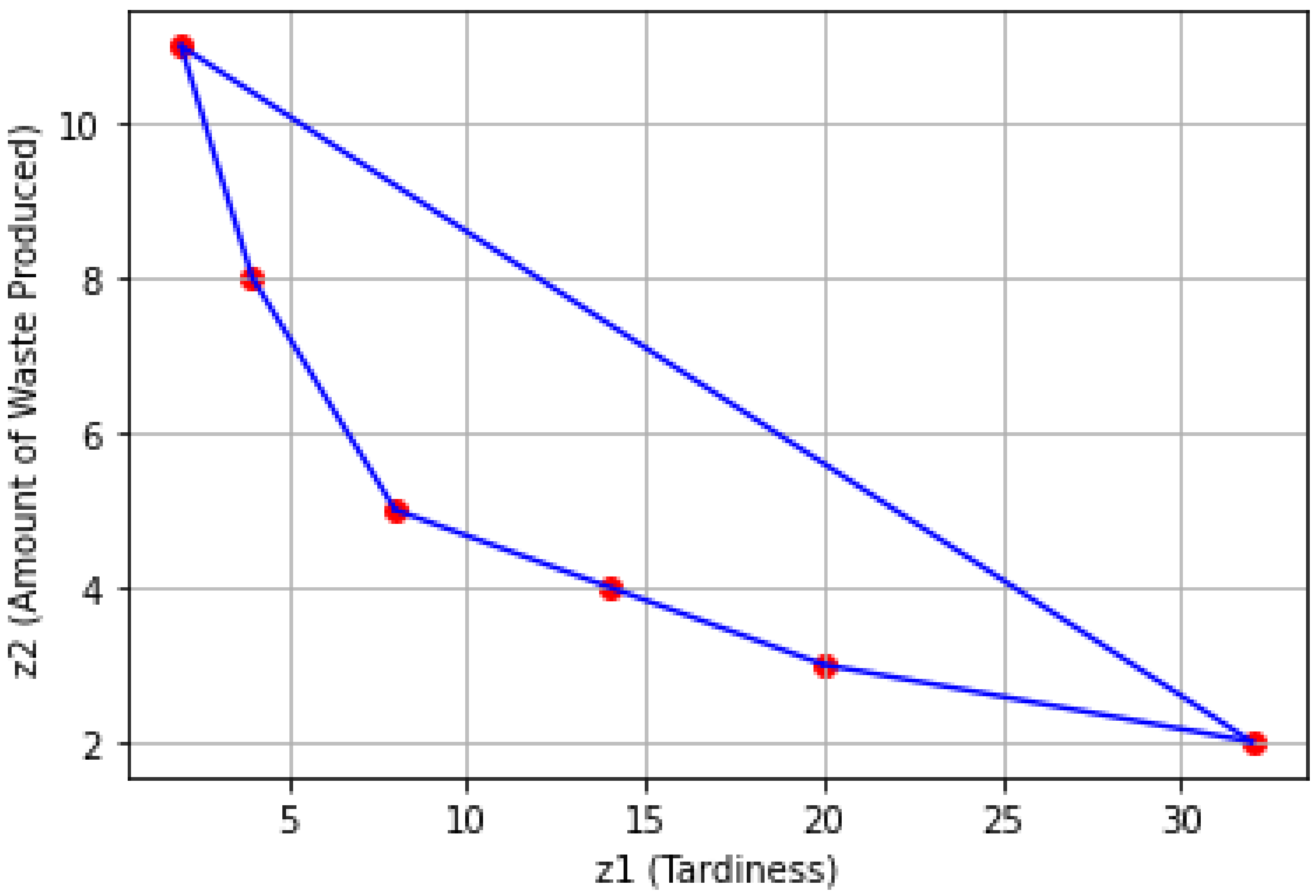


Nondominated points

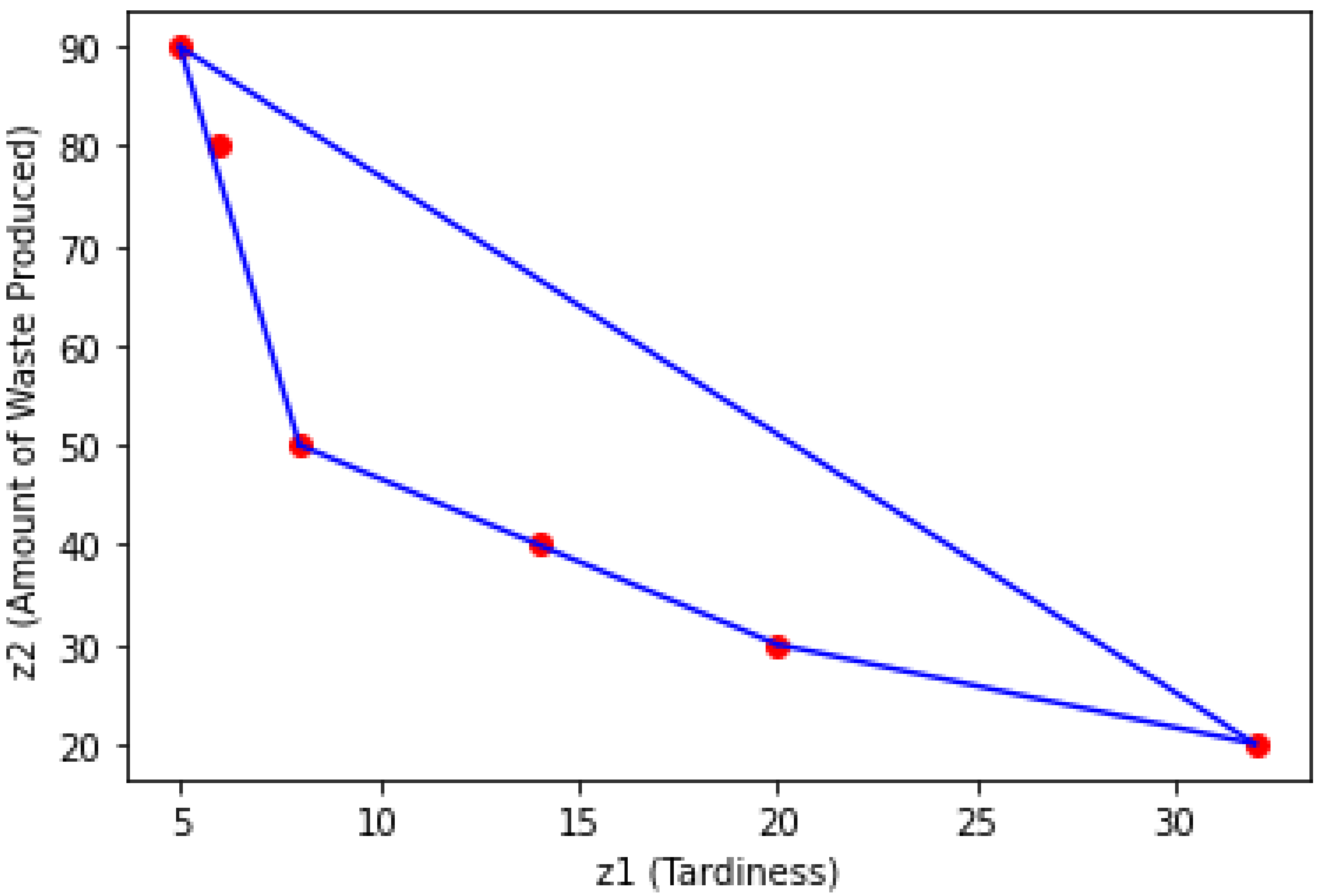


Pareto Frontier

Findings for the ε-Constraint Algorithm



Convex hull



Convex Hull Indicating an Unsupported Point for a Different Scenario

Outputs

Machine 1	Machine 2	Machine 3
Job 3	Job 5 Job 4	Job 1 Job 2

epsilon = 32
tardiness = 32
# of defective product = 2

Machine 1	Machine 2	Machine 3
Job 3 Job 4	Job 1 Job 2	Job 5 Job 4

epsilon = 20
tardiness = 20
# of defective product = 3

Machine 1	Machine 2	Machine 3
Job 5 Job 4	Job 3 Job 2	Job 1 Job 4

epsilon = 14
tardiness = 14
# of defective product = 4

Machine 1	Machine 2	Machine 3
Job1 Job 3	Job 4 Job 2	Job 5 Job 4

epsilon = 8
tardiness = 8
# of defective product = 5

Machine 1	Machine 2	Machine 3
Job 4	Job 4 Job 1 Job 3	Job 5 Job 4 Job 2

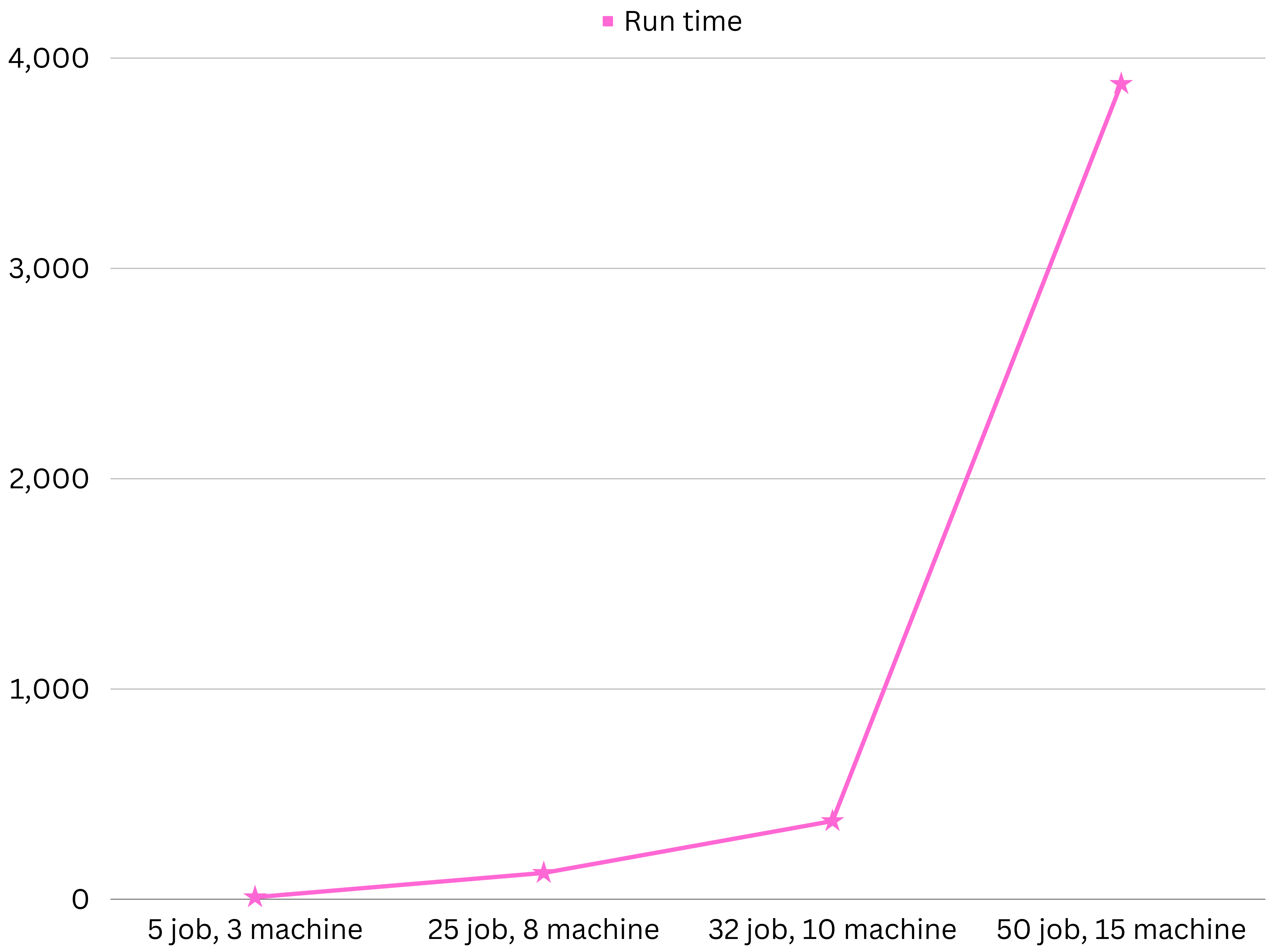
epsilon = 4
tardiness = 4
# of defective product = 8

Machine 1	Machine 2	Machine 3
Job 4 Job 1 Job 3	Job 4 Job 2	Job 4 Job 5

epsilon = 2
tardiness = 2
# of defective product = 11

Computational Analysis

Number of jobs	Number of machines	Run time (seconds)
5	3	10.56
25	8	125.97
32	10	372.6
50	15	3877.8



References

[1] Khodakaram GSalimifard, Jingpeng Li, Davood Mohammadi, and Reza Moghdani.
A multi objective volleyball premier league algorithm for green scheduling identical
parallel machines with splitting jobs. Applied Intelligence, 51(7):4143–4161, 2021

Thank You