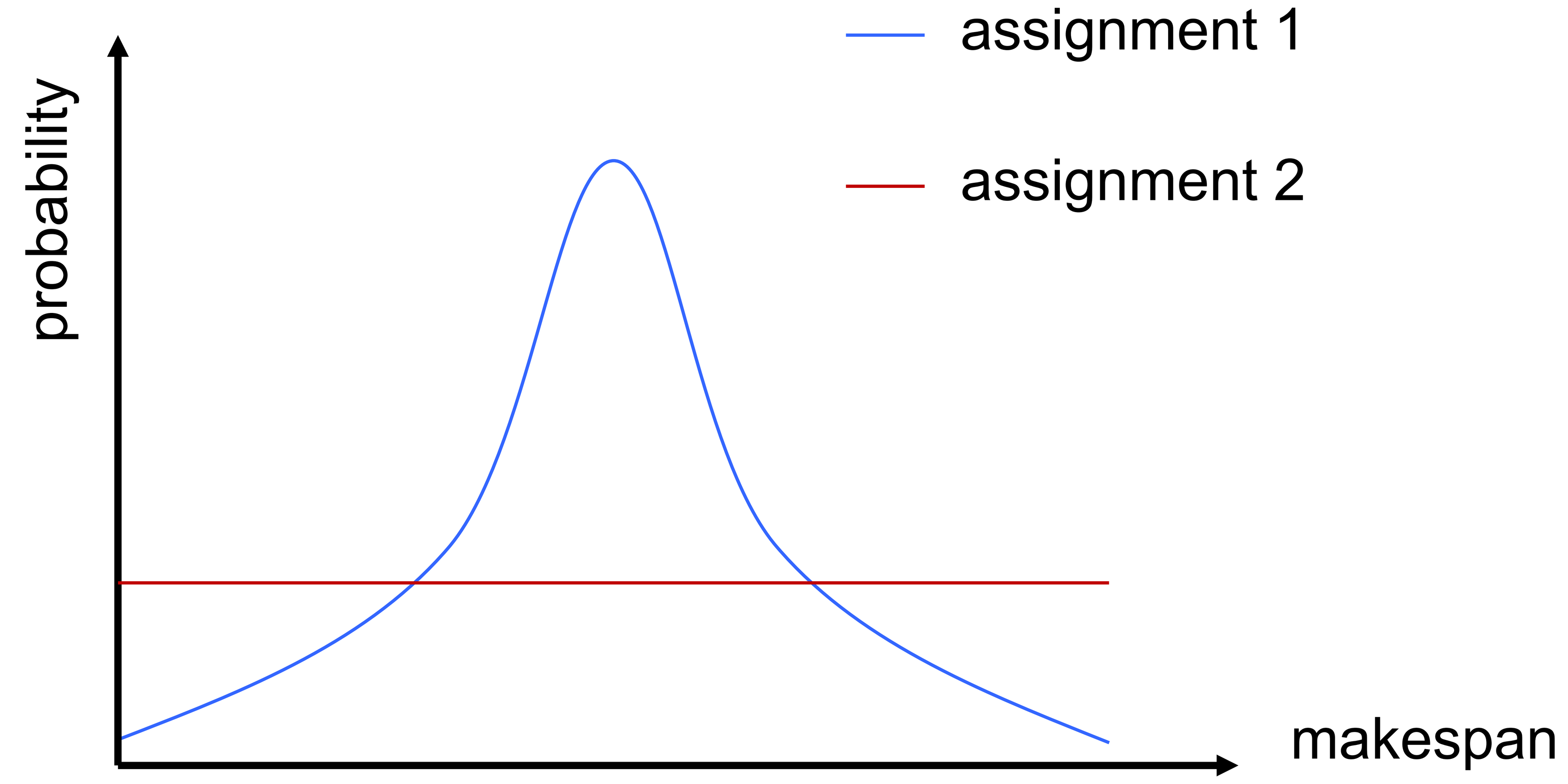
Scheduling Stochastic Jobs - Complexity and Approximation Algorithms

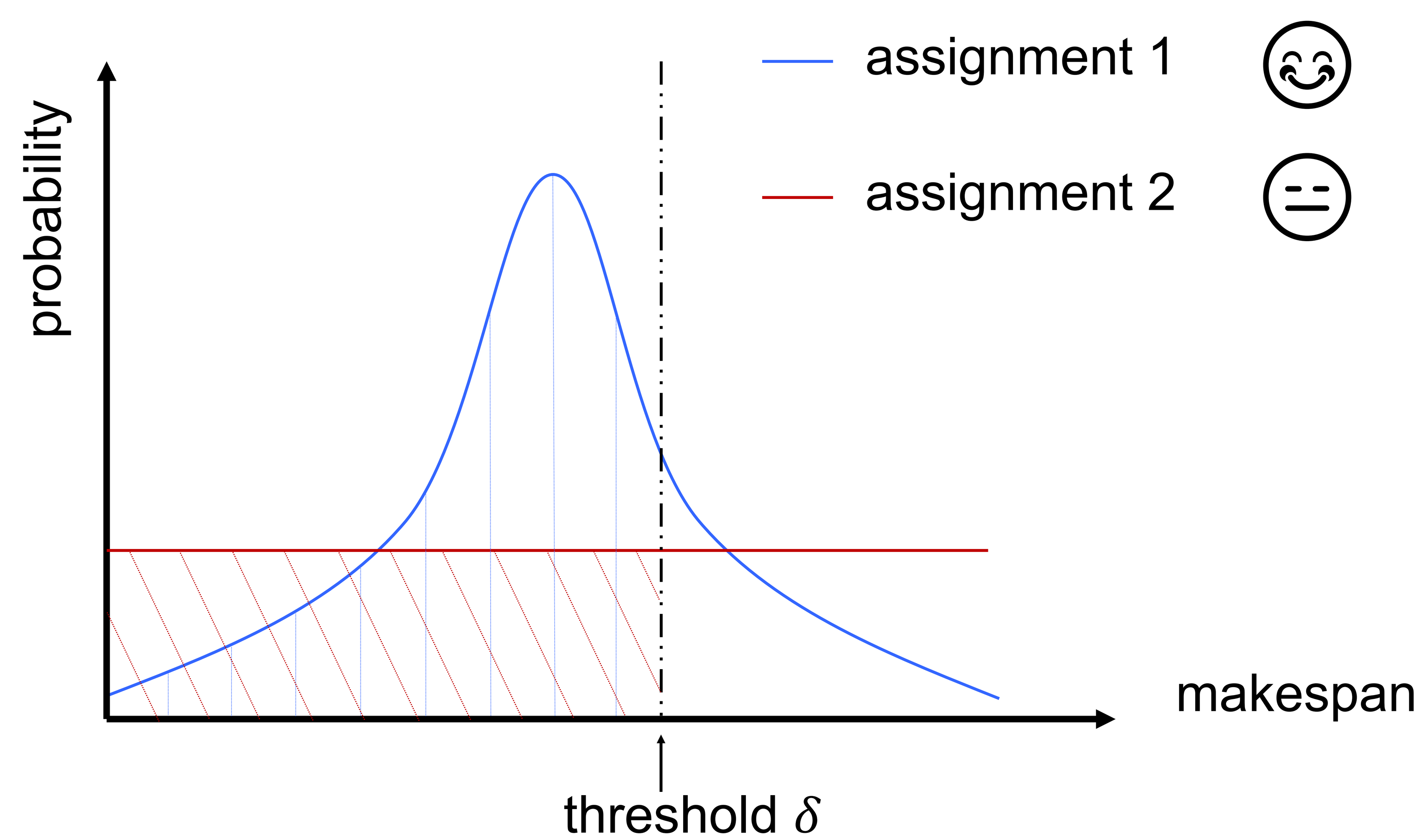
Liangde Tao[1], Lin Chen[2], Guochuan Zhang[1]
vast.tld@gmail.com, chenlin198662@gmail.com, zgc@zju.edu.cn
Zhejiang University, China[1] Texas Tech University, United State[2]

Motivation

Scheduling is a fundamental problem in computer science and has received extensive study in the literature. In the real world, the processing time of job is not fixed, it could change due to various factors. Minimizing the expectation of the makespan is a widely used objective in stochastic scheduling. However, it may not give a robust assignment. See the following example.



To overcome this, we adopt the β-robust measure [Daniels and Carrillo, 1997] as the objective. In stochastic scheduling, we can not guarantee the makespan is always no larger than the given threshold. But we can maximize the probability that the makespan is no larger than the given threshold. In another word, given the threshold δ, we want to find the assignment which maximizes the probability that the makespan is not larger than δ.



Problem Statement

The β-robust Scheduling Problem (β-RSP)

- Given a set of identical machines, a set of jobs whose processing times follow a normal distribution and a common due date
- The goal is to assign jobs to machines such that the probability that all the jobs are completed by the given common due date is maximized.

Our Contributions

- **Theorem** [Hardness]

For any $c \in (0,1)$, there is no polynomial time algorithm that can return a feasible solution with objective value at least $OPT - c$ for any instance of the β-RSP, assuming $\mathrm{P} \neq NP$.

- **Theorem** [Algorithm-a]

For an arbitrary small $\epsilon > 0$, there exists an algorithm gives an $OPT - \epsilon$ solution for the β-RSP within $O\left(\left(\frac{n^2 mL}{\epsilon}\mu_{max}\right)^m\right)$ time.

- **Theorem** [Algorithm-b]

For an arbitrary small $\epsilon > 0$, there exists an algorithm gives an $OPT - \epsilon$ solution for the β-RSP within $(L/\epsilon)^{O(m)}(mk)^{O(mk)}$ time.

Technique Overview

Hardness: utilize the NP-hardness of *2-Partition*

Algorithm-a: dynamic programming + trim state space

Algorithm-b: guess two ranges for each machine (the summation of means and the summation of variances of jobs on this machine); transform the nonlinear integer linear program to a series of integer linear programming feasibility problems.

$$\max \prod_{i\in M} \Phi\left(\frac{\delta - \mu_{M_i}}{\sqrt{\sigma^2_{M_i}}}\right)$$

$$\begin{array}{llll} \text{s.t.} & \sum_{j\in J} \mu_j \cdot x_{ij} = \mu_{M_i} & \forall i \in M & \\ & \sum_{j\in J} \sigma_j^2 \cdot x_{ij} = \sigma^2_{M_i} & \forall i \in M & \\ & \sum_{i\in M} x_{ij} = 1 & \forall j \in J & \\ & x_{ij} \in \{0,1\} & \forall i \in M, j \in J & \end{array} \quad \rightarrow \quad \begin{array}{ll} \sum_{j\in J} \mu_j \cdot x_{ij} \in S_{\mu_i} & \forall i \in M \\ \sum_{j\in J} \sigma_j^2 \cdot x_{ij} \in S_{\sigma_i^2} & \forall i \in M \\ \sum_{i\in M} x_{i\ell} = n_\ell & \forall \ell \in K \\ x_{i\ell} \in \mathbb{N} & \forall i \in M, \ell \in K \end{array}$$