Appendix

A Proof of Nodes Specification Covering All Nodes

We prove the following theorem which shows a finite set of p number of pipeline templates, where the number of nodes is $(n_0, n_1, \ldots, n_{p-1})$ $(n_i < n_{i+1})$, can fully cover the node cluster with feasible number of nodes N' any time irrespective of how many failures happen on the cluster as long as its feasibility holds.

Theorem A.1. *N' nodes ($(f+1)n_0 \leq N' \leq N$) can always be represented as a linear combination of the p pipeline templates with $(n_0, n_1, \ldots, n_{p-1})$ number of nodes, respectively, if the following two conditions are satisfied:*

1. $p > n_0 - 1$.
2. *n_i are consecutive integers ($n_i + 1 = n_{i+1}$).*

Proof. We first formulate an integer linear combination to represent N':

$$N' = x_0 n_0 + x_1 n_1 + \ldots, x_{p-1} n_{p-1} \tag{7}$$

where x_i is the number of pipelines to be instantiated from the pipeline template with n_i number of nodes.

The Frobenius number $g(n_0, n_1, \ldots, n_{p-1})$, the largest number that cannot be represented as a linear combination of Equation 7, has proven to be:

$$g = \left(\left\lfloor \frac{n_0 - 2}{p - 1} \right\rfloor \right) + d(n_0 - 1) \tag{8}$$

if the integer set $(n_0, n_1, \ldots, n_{p-1})$ is an arithmetic sequence, i.e., $n_i = n_0 + d(i-1)$ [39].

When we apply both Requirements 1 and 2, $g = n_0 - 1$. Fault tolerance threshold f is a non-negative integer; the minimum feasible number of nodes $N' = (f+1)n_0$ is n_0. Therefore, any feasible N' that is larger than g and can be represented as a linear combination of Equation 7. □

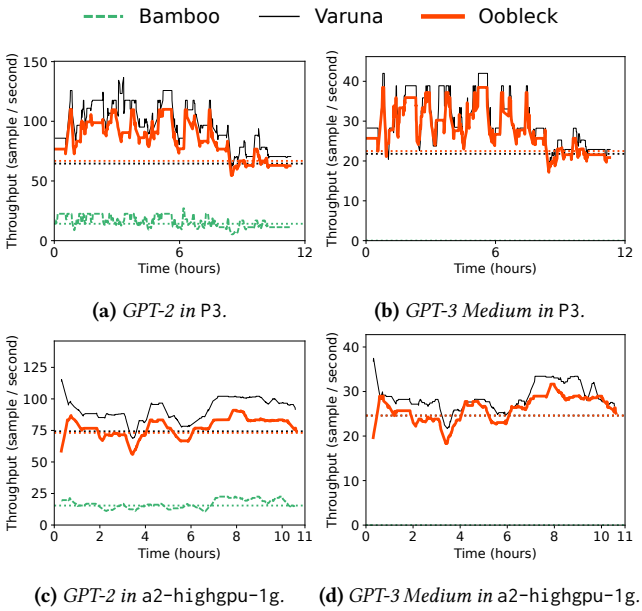
B Proof of Guarantee for Pipeline Template Availability When Merging Pipelines

We first show this when failures happen in a single pipeline. When we lose k nodes ($k > 0$) from a pipeline, where all pipelines have n_0 nodes and are not able to yield any node, Oobleck instantiates a new pipeline with $2n_0 - k$ nodes by merging it with another n_0-node pipeline. We prove that a pipeline template with $2n_0 - k$ nodes is always available.

Theorem B.1. *A set of pipeline templates always includes a pipeline template with $2n_0 - k$ nodes ($2n_0 - k \geq n_0$).*

Proof. A set of pipeline templates has pipeline templates with up to $N - fn_0$ nodes (§4.1.1). Assume that we do not have a pipeline template with $2n_0 - k$ number of nodes specification, then $N - fn_0 < 2n_0 - k$ and $N < (f+2)n_0 - k$ are assumed to be true. To not break the fault tolerance threshold that we maintain at least $f+1$ model replicas after merging two pipelines, we must have at least $f+2$ replicas, i.e., we should have had at least $(f+2)n_0$ nodes before failures. Since the initial number of nodes N is always larger than the number of currently remaining nodes, $N > (f+2)n_0$ inequality holds and it contradicts our initial assumption. Therefore, we have a pipeline template with $2n_0 - k$ number of nodes. □

When failures happen across several pipelines, multiple pipelines can have less than n_0 nodes. Oobleck repeatedly merges two pipelines until a new pipeline has enough number of nodes. Assume we merged m pipelines to get enough number of nodes, i.e., $\sum_{i=0}^{m} n_{p_i} \geq n_0$. It means merging $m-1$ pipelines was not enough to get n_0 nodes, i.e., $\sum_{i=0}^{m-1} n_{p_i} < n_0$. With $n_{p_m} \leq n_0$, we have an inequality $n_0 \leq \sum_{i=0}^{m-1} n_{p_i} + n_{p_m} < 2n_0$. It has already been proved by Theorem B.1 that we have a pipeline template for all numbers in the range.



(a) *GPT-2 in* P3. **(b)** *GPT-3 Medium in* P3.

(c) *GPT-2 in* a2-highgpu-1g. **(d)** *GPT-3 Medium in* a2-highgpu-1g.

Figure 12. *GPT-2 and GPT-3 medium throughput changes in Amazon EC2* P3 *(top) and Google* a2-highgpu-1g *(bottom) instances.*

C Throughput of All Models in Spot Instances

Figure 12 shows throughput of unpresented models in the paper due to lack of space, running on Amazon EC2 P3 spot instances and Google a2-highgpu-1g spot instances. Varuna could avoid fallback overhead by successfully checkpointing ahead of preemption in small models, (e.g., BERT-large, GPT-2, and GPT-3 medium), thus Varuna throughput matches Oobleck on average. However, in large models, Oobleck outperforms it. Note that lines are smoothed for visibility and do not precisely represent throughput. Varuna has more spikes down to 0 throughput in reality thus has less throughput.