## E. Failures and their implications

这小节我们将对失效节点进行分析，因为节点状态和其运行的Job密切相关，因此我们将从失效节点和失效job两方面进行研究。

1. **失效节点统计：**

在统计的1313节点中，只有大约1.5%（20个）节点在任意时间点都无失效情况，剩余98.5%（1293个）的节点都存在一定数量的失效时间点。每个节点的失效次数分布如图1所示。

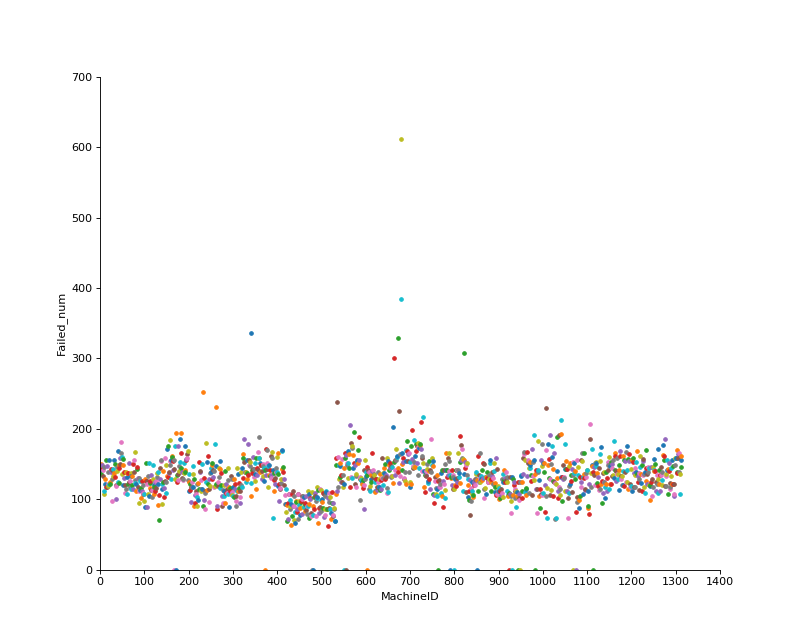


Figure 1 节点失效次数分布

**失效job统计：**

为了更加清楚地分析节点失效的原因，我们对job的失效情况也进行了统计。这些节点一共运行了12591种类型的job，我们统计了这12591类型job的各自失效的次数（这数据是不完全精确的，因为出于隐私，Alibaba隐藏了部分数据信息）。根据统计结果，大概82%的job是完全未失效的，剩余的job都存在一定数量的失效次数。每种job的失效次数分布如图2所示。

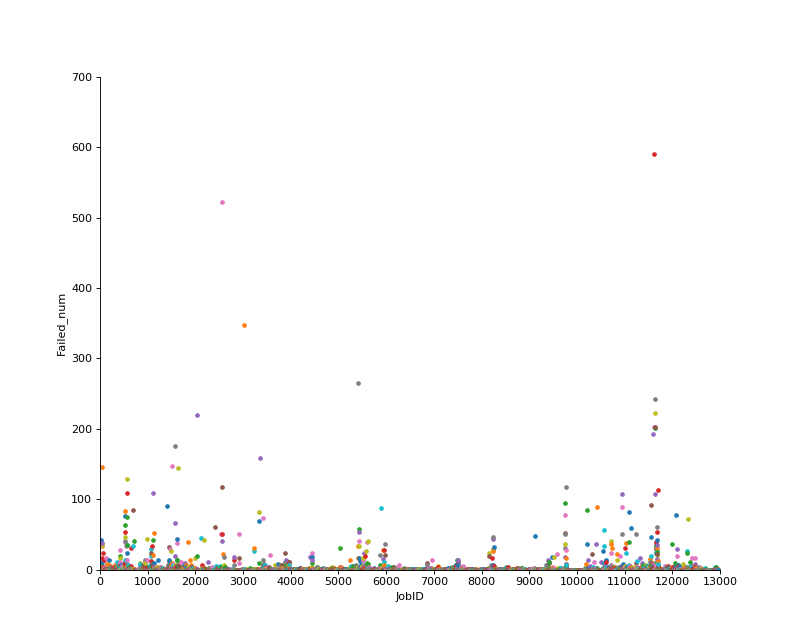


Figure 2 job失效次数分布

1. **失效节点分布：**

节点在不同时间失效分布是不一致的，为了描述它们的关系，我们对节点随时间变化的失效情况进行了分析。图3描述了随时间变化，失效节点的分布。图4描述了随时间的变化，失效节点个数的分布。

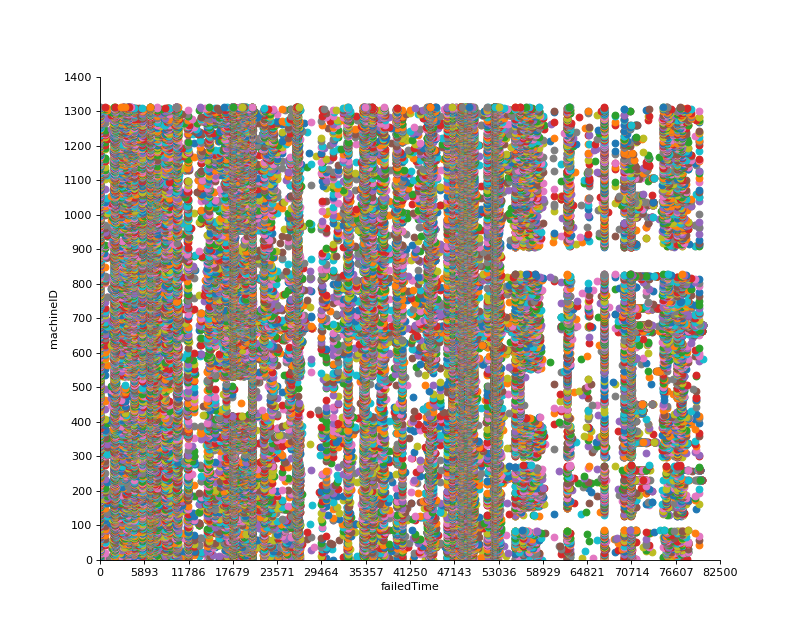


Figure 3 不同时间点失效节点的分布

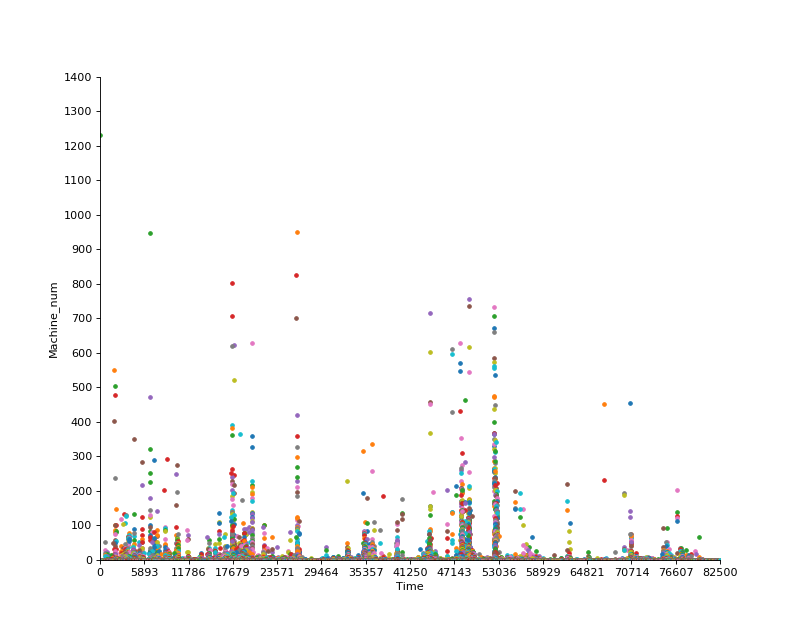


Figure 4 不同时间点失效节点次数的分布

**失效Job分布：**

同样，我们对Job随时间变化的失效情况也进行了分析。图5描述了随时间变化，失效Job的分布。图6描述了随时间的变化，失效Job个数的分布。

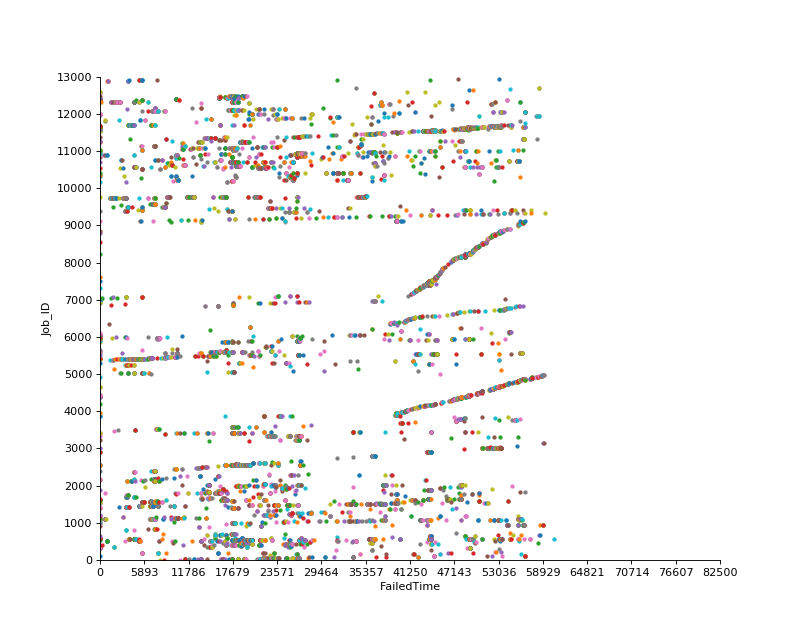


Figure 5 不同时间点失效Job的分布

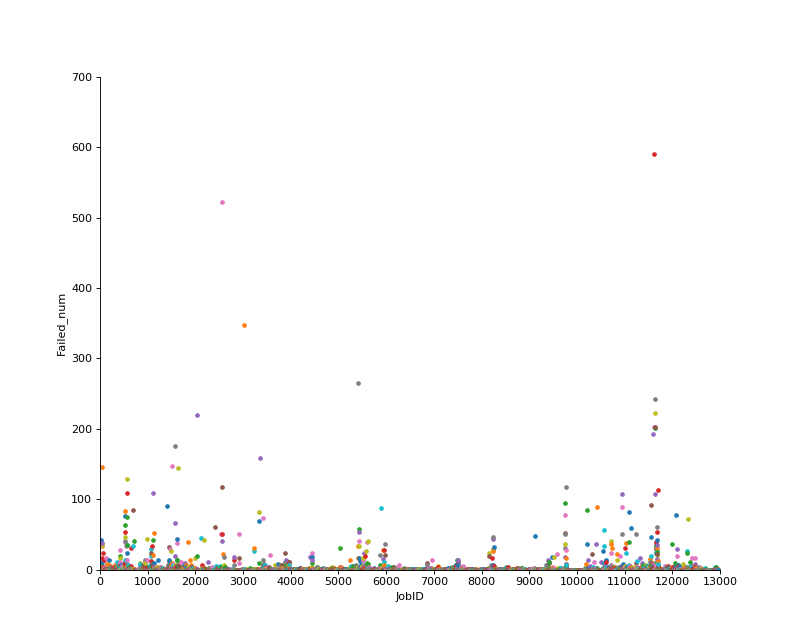


Figure 6 不同时间点失效Job次数的分布

1. **失效节点分析**

这一节我们对上两节的结果进行分析，我们可以发现对于某一些节点（例如：679,680,341,673）的失效次数相比其它的节点，失效次数明显更多。而只有少数节点（例如：167，478，1075）完全没有失效。为了分析原因，我们研究了失效节点和未失效节点的cpu和momery 的capacity、利用率以及在其上运行的Job类型。为了确认节点失效是由上述哪种原因造成的，我们进一步对失效Job的分布进行了分析。同失效节点分布类似，失效job分布是不均匀的，部分Job（例如：11624，2567,3012）的失效次数远大于其他的一些Job，对于失效次数比较多的Job，我们可以对该类Job的特点进行记录，分析其失效的原因。经上述分析，我们猜测节点在运行那些失效次数较多的Job时更容易失效，而在运行较不易失效的Job时可能由于节点自身的cpu和memory的限制而成为失效节点。

为了更加准确的评估节点失效的时间，从而预估节点的维护时间，我们计算了每个节点的MTBF（Mean Time Between Failure）。具体情况如图7所示。

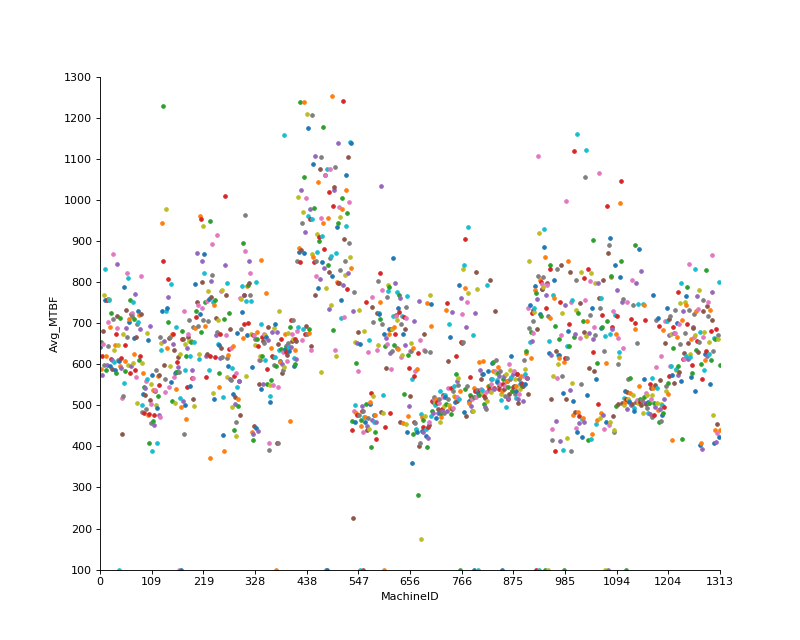


Figure 7 不同节点的MTBF

从图7我们可以发现，大多数的节点的MTBF集中在2000s~4000s之间，少数节点在4000s~7000s之间，还有极少数的节点低于2000s。

1. **失效节点预测和改善**

根据以上对失效节点分布情况的描述以及分析，我们可以根据节点id，该节点上运行Job类型以及在哪个时间开始运行来预测节点大致的状态，若预测的状态是Failed，我们可以将Job迁移到别的节点上或者等待下一个时间点再运行，以降低失效的可能性。除此之外，我们还可以根据每个节点的MTBF以及上一个失效时刻来预测下一个失效时刻，从而制定最佳的一个维修计划，延长节点连续运行的时间。

为了降低节点失效的可能性，我们还可对失效率比较高的Job类型进行分析，确定其失效的可能原因，节点在运行该job时，避免可能的失效原因，最大限度的降低失效可能性。针对那些失效率低的节点，可将在其他节点运行的Job迁移到这些节点上面，减少其他节点的失效次数。

我们将MTBF比较小的节点称为高频率故障节点，针对这些节点，根据它们的特征，可改善其维修技术，以提高它们的MTBF。根据每个节点的MTBF，我们也可合理分配维修工作，从而更好的对节点进行维护。