**Concurrency vs. Parallelism**

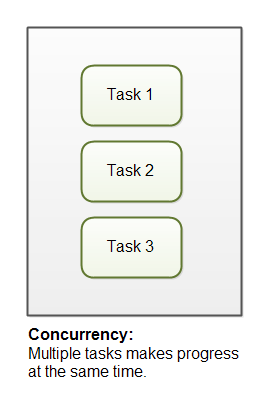
* [Concurrency](http://tutorials.jenkov.com/java-concurrency/concurrency-vs-parallelism.html#concurrency)
* [Parallelism](http://tutorials.jenkov.com/java-concurrency/concurrency-vs-parallelism.html#parallelism)
* [Concurrency vs. Parallelism In Detail](http://tutorials.jenkov.com/java-concurrency/concurrency-vs-parallelism.html#concurrency-vs-parallelism)

词条并发和平行化经常提到与多线程程序相关。但是并发和平行化具体什么意思，并且它们是不是相同的词条？

简单回答是“不是”。它们不是相同的词条，尽管它们表面看来确实相似。最终发现和理解并发和平行化的不同点也花费了我一些时间。因此我决定增加一篇关于并发vs平行化的文章到Java并发教程中。

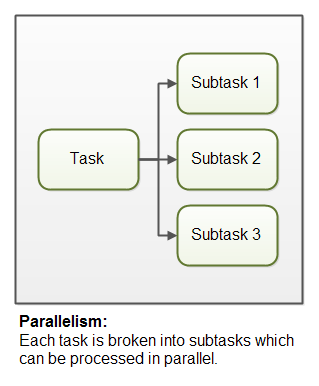
**Concurrency**

并发意味着应用同一时间(并发地)处理多个任务。如果电脑只有一个CPU那么应用可能不能在精确的同一时间处理超过一个任务，但是在应用中超过一个任务在某一段时刻被处理。它并不完全完成一个任务，在开始下一个任务之前。



**Parallelism**

平行化意味着一个应用划分任务到更小的子任务，它们可以被平行执行，例如在多个CPU在精确的同一时刻。



**Concurrency vs. Parallelism In Detail**

正如你所见，并发关联于一个应用如何处理它工作的多任务。一个应用可能在一个时间处理一个任务(顺序)或者在同一时刻工作在多个任务上(并发)。

另一方面，平行化，与一个应用如何处理每个独立的任务有关。一个应用可能序列地从开始到结尾处理任务，或者划分任务成可以平行完成的子任务。

正如所见，一个应用可以是并发的，但不是平行化的。这意味着它同时处理超过一个任务，但任务不能分解成为子任务。

一个应用也可能是平行化但不是并发的。这意味着应用在一个时间只工作在一个任务上，任务可以被分解成子任务，这些子任务可以平行化处理。

另外，一个应用可以既不是并发也不是平行化。这意味着它一个时刻只工作在一个任务上，任务不能分解为子任务用于平行化执行。

最后，一个应用也可以是并发和平行化的，表示它同时工作在多任务上，并且分解每个任务为子任务用于平行化执行。然而，并发和平行化的一些好处在这种情景下可能会丢失，如果电脑中的CPU已经保持相当繁忙单独处理并发或平行化。融合可能导致只有一小点性能增益或者甚至性能损失。确保在你盲目地采用并发平行模型之前进行分析测量。