Streaming是建立在Spark上的实时计算框架，提供了丰富的API，以支持Spark处理大规模流式数据。

Streaming窗口操作

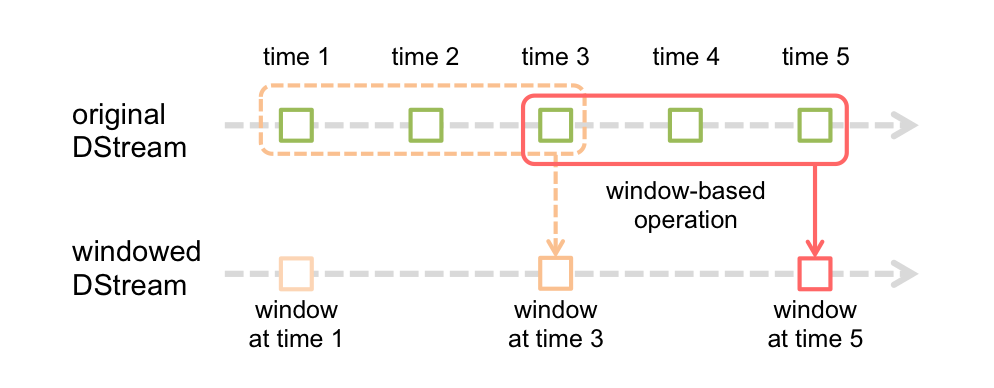
Streaming是按batch Interval来划分job的，但有时需要对某个时刻开始的某个指标进行持续跟踪和计算，并更新该指标（例如当天网页的点击次数或者本月的最新销售额等），那么这样就需要根据此前的状态结果和新batch Interval 的数据计算出新的状态结果。



窗口操作：周期性的对过去固定长度的最近时间段内的数据进行处理。

我们把这个时间看成是一个滑动的窗口，因为时间段固定(batch Interval), 而操作的起始时刻就是在等时后延的。

窗口操作设计了两个参数，一个是滑动窗口的参数(window Interval)，另一个是窗口滑动的频率(slide Interval)，这两个参数必须是batch Interval的整倍数。



|  |  |
| --- | --- |
| **Transformation** | **Meaning** |
| **window**(*windowLength*, *slideInterval*) | 返回一个基于源Dstream的窗口批次计算后得到的新的Dstream. |
| **countByWindow**(*windowLength*, *slideInterval*) | 返回基于滑动窗口的DStream中的元素的数量. |
| **reduceByWindow**(*func*, *windowLength*, *slideInterval*) | 基于滑动窗口对源DStream中的元素进行聚合操作，得到一个新的DStream. |
| **reduceByKeyAndWindow**(*func*, *windowLength*, *slideInterval*, [*numTasks*]) | 基于滑动窗口对键值对(K, V)类型的DStream中的值按K使用聚合函数func进行聚合操作，得到一个新的DStream. |
| **reduceByKeyAndWindow**(*func*, *invFunc*, *windowLength*, *slideInterval*, [*numTasks*]) | 一个更高效的reduceByKeyAndWindow的实现版本，现对滑动窗口新的时间间隔内数据增量聚合并移去最早的与新增数据量的时间间隔内的数据统计量. 例如，计算t+4秒这个时刻过去5秒窗口的WordCount,那么可以将t+3时刻过去5秒的统计量加上[t+3,t+4]的统计量，在减去[t-2,t-1]的统计量，这种方法可以服用中间3秒的统计量，提高统计的效率 |
| **countByValueAndWindow**(*windowLength*, *slideInterval*, [*numTasks*]) | 基于滑动窗口计算DStream中每个RDD内每个元素出现的频次并返回DStream[(K,Long)]，其中K是RDD中元素的类型，Long是元素频次，countByValue一样，reduce任务的数量可以通过一个可选参数进行配置. |

和其他DStream一样，WindowedDStream的生成需要有自己的compute方法