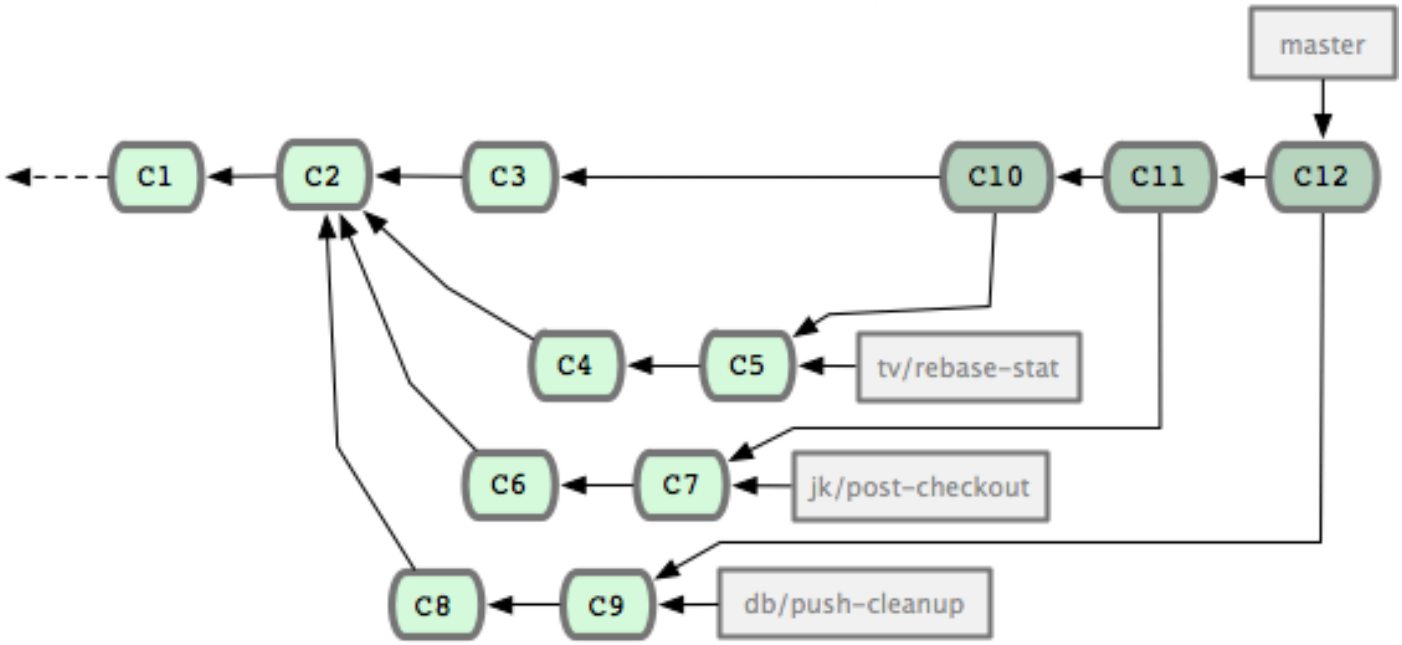
[译]Git回滚合并

May 26, 2015

如果工作流严重依赖分支合并的话，也难免会碰到需要回滚合并的情况。而且回滚还分为两种情况：永久回滚，或者回滚后在稍后重新合并。

假设有如下分支图：



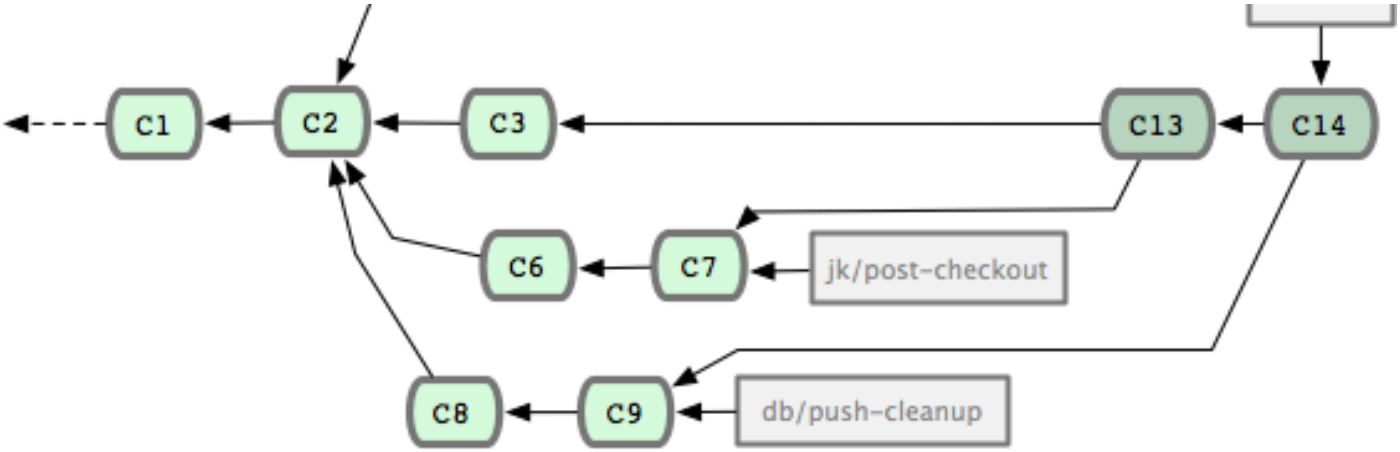
> 注：Git分支图中的箭头表示依赖关系，并不是分支发展路线。发展路线和箭头是相反的。也就是图中是从C1开始一直发展到C12的。

假设要回滚C10。

第一种解决方案是将 `master` 回退到C8，然后将两个特性分支 `jk/post-checkout` 和 `db/push-cleanup` 合并过来。

```
git checkout master
git reset --hard [sha_of_C8]
git merge jk/post-checkout
git merge db/push-cleanup
```

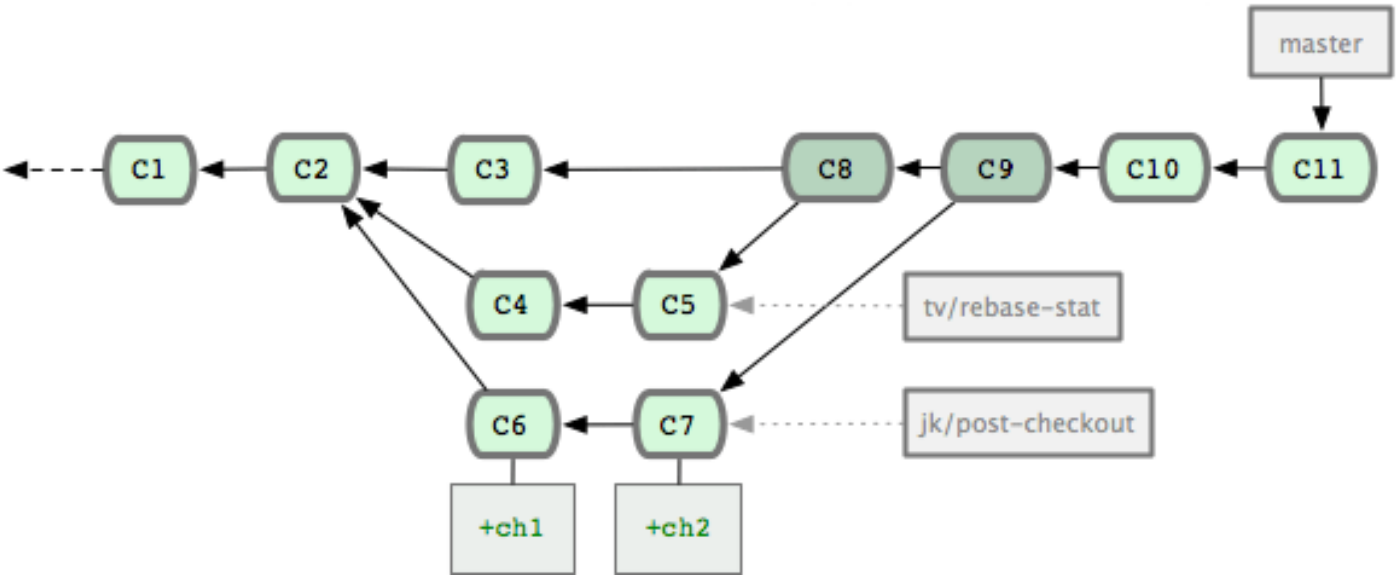
完成之后，分支图如下：



接下来就可以继续在新的 `master` 上工作，然后在适当的时候将 `tv/rebase-stat` 合并回来。

回滚合并

如果在很久之后才发现要回滚，或者其它人已经在合并之后提交了代码，分支图会是这样：

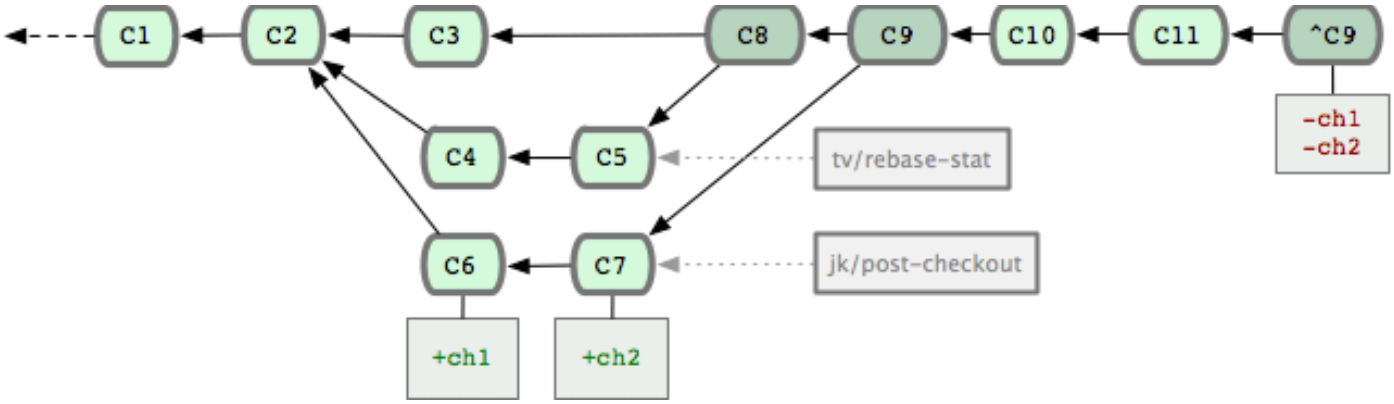


这种情况下要么回退一次合并，要么退回去，再重新合并，然后将新的变更（C9和C10）cherry-pick过来。后者容易让人迷惑，做起来也不容易，尤其是在合并之后提交很多的情况下。

`git revert` 能很好地处理合并的回退。你需要指定需要回退的那次合并提交记录，并指定保留合并中的哪一条线（parent）。假设我们需要回退合并 `jk/post-checkout` 的记录，则这样做：

```
git revert -m 1 [sha_of_C8]
Finished one revert.
[master 88edd6d] Revert "Merge branch 'jk/post-checkout'"
 1 files changed, 0 insertions(+), 2 deletions(-)
```

完成后会产生一个新提交，这个提交回滚了合并过来的内容。其结果和一个包含被合并过来分支改动的cherry-pick（反向操作，即回滚）差不多。

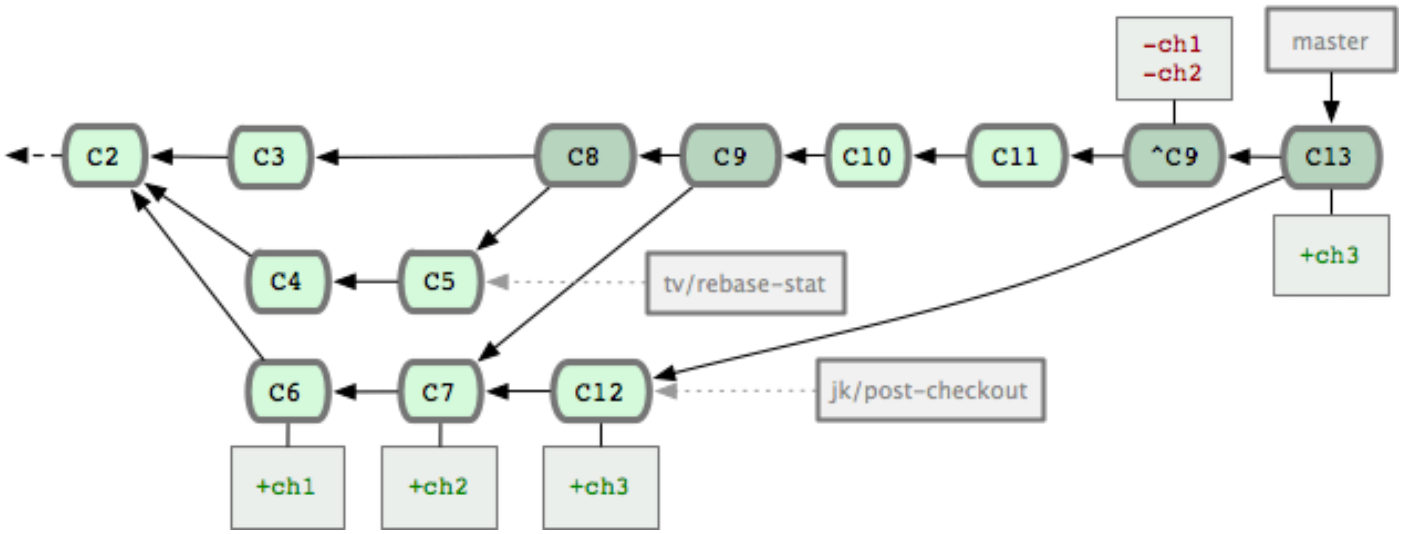


回滚“回滚”

假设在回滚之后，我们需要再次合并这个分支。如果你直接合并的话，什么都不会发生。

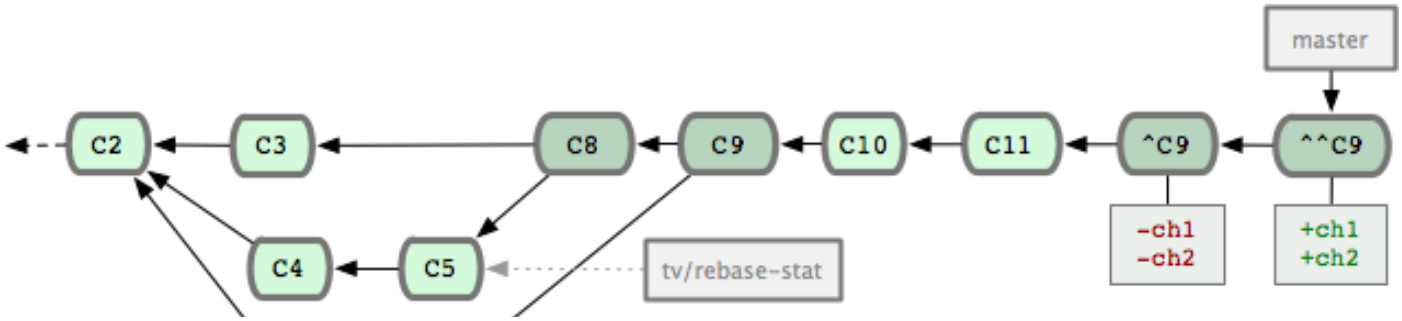
```
git merge jk/post-checkout
Already up-to-date.
```

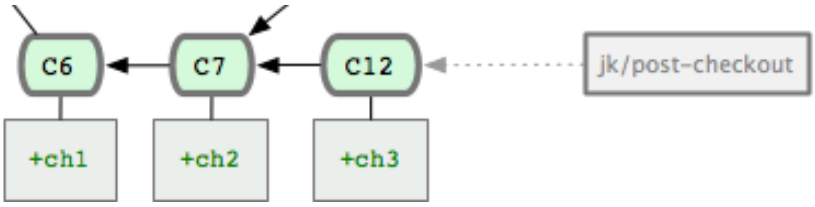
更令人困惑的是，如果你回到分支，再做一些修改后再合并，则只有新产生的修改会被合并过来。



这种状态是一种非常奇怪的状态，有可能导致冲突或者难以理解的错误。此时你真正想做的事情应该是回滚“上一次对合并的回滚操作”。

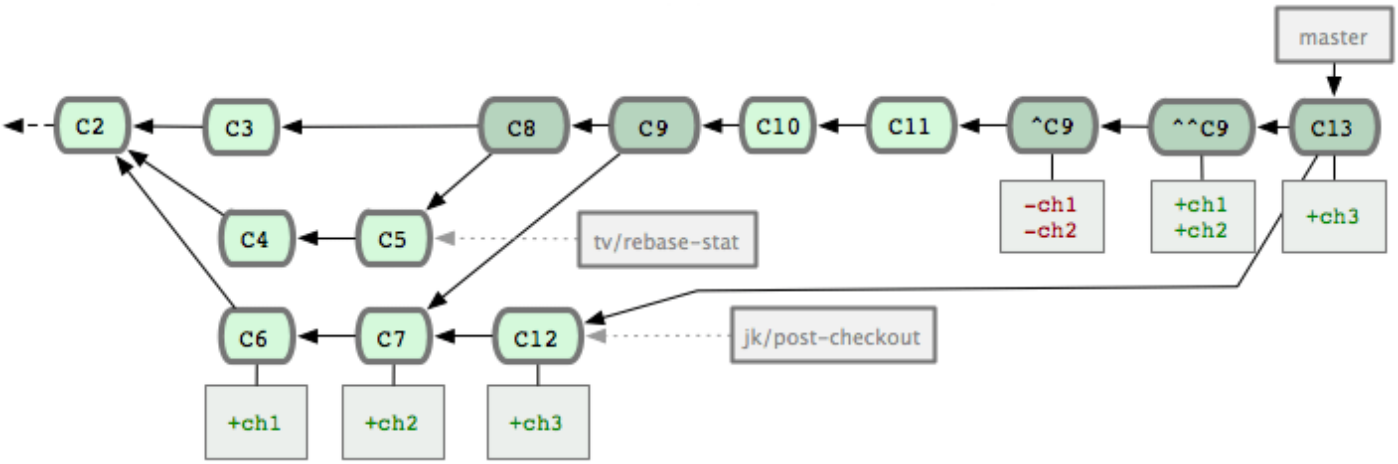
```
git revert 88edd6d
Finished one revert.
[master 268e243] Revert "Revert "Merge branch 'jk/post-checkout'""
 1 files changed, 2 insertions(+), 0 deletions(-)
```





现在我们将分支恢复到了合并之后的情况，如果分支上有新的改动，就可以直接合并了。

```
git merge jk/post-checkout
Auto-merging test.txt
Merge made by recursive.
 test.txt |    1 +
 1 files changed, 1 insertions(+), 0 deletions(-)
```



最后，建议使用 `git merge --no-ff` 合并分支，这样可以保持 分支不是快进的，使它可以回滚。

> 原文地址https://git-scm.com/blog/2010/03/02/undoing-merges.html，本文并未逐字逐句翻译，仅根据理解摘录了重要部分。