redhat.

05 Gitshop

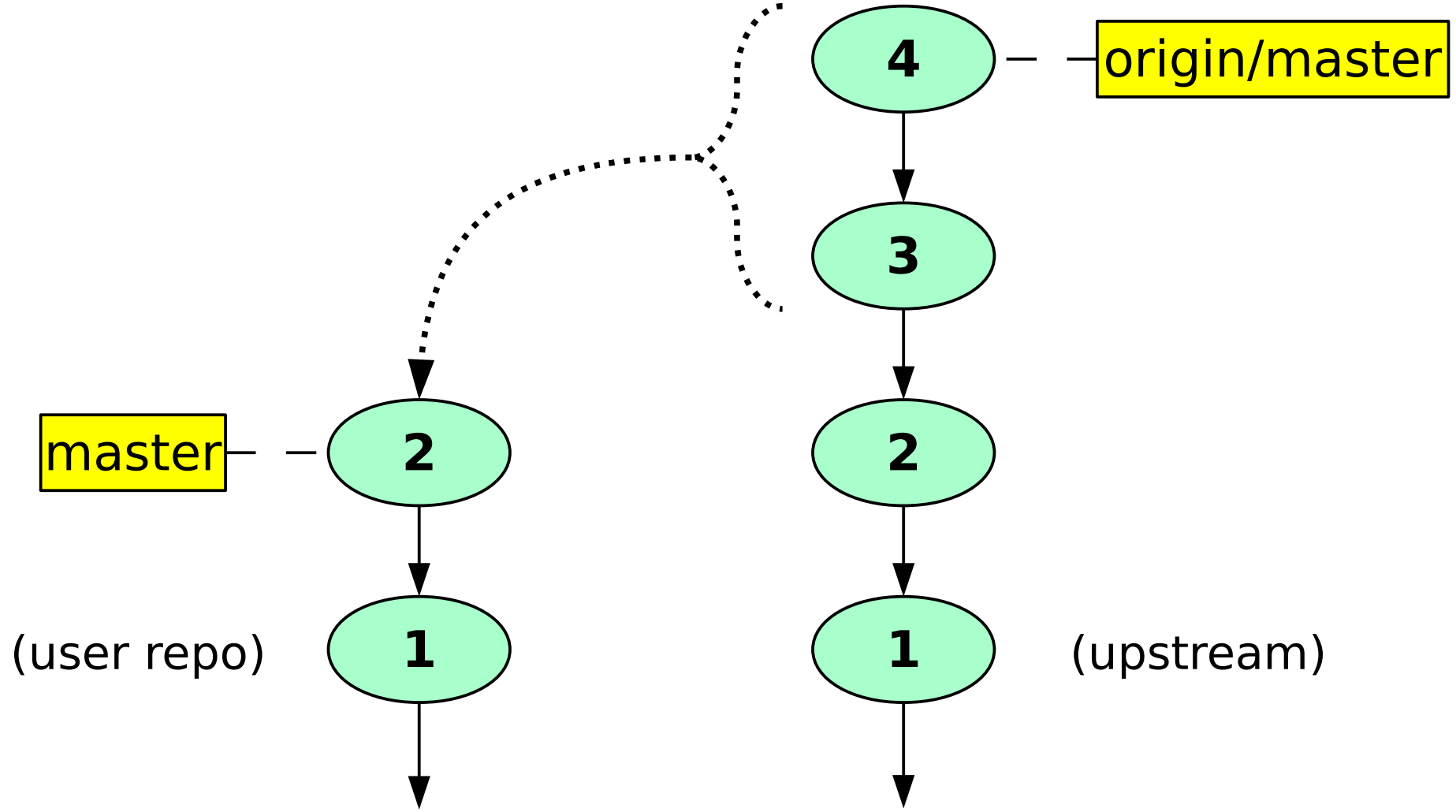
Git - merge

Jiří Jabůrek (jjaburek)
Red Hat 2012

Degenerate merges

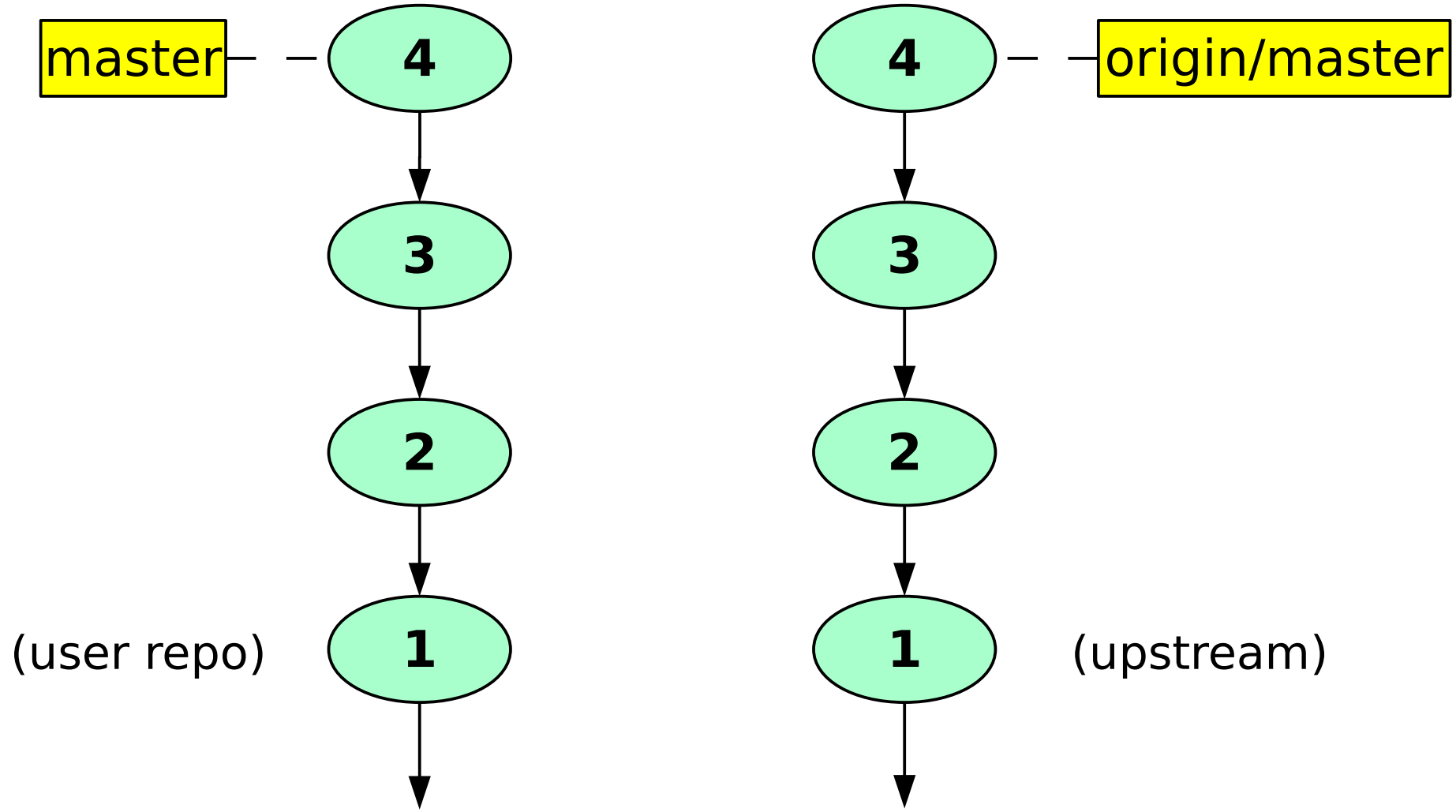
Fast-forward merge

```
$ git merge origin/master
```



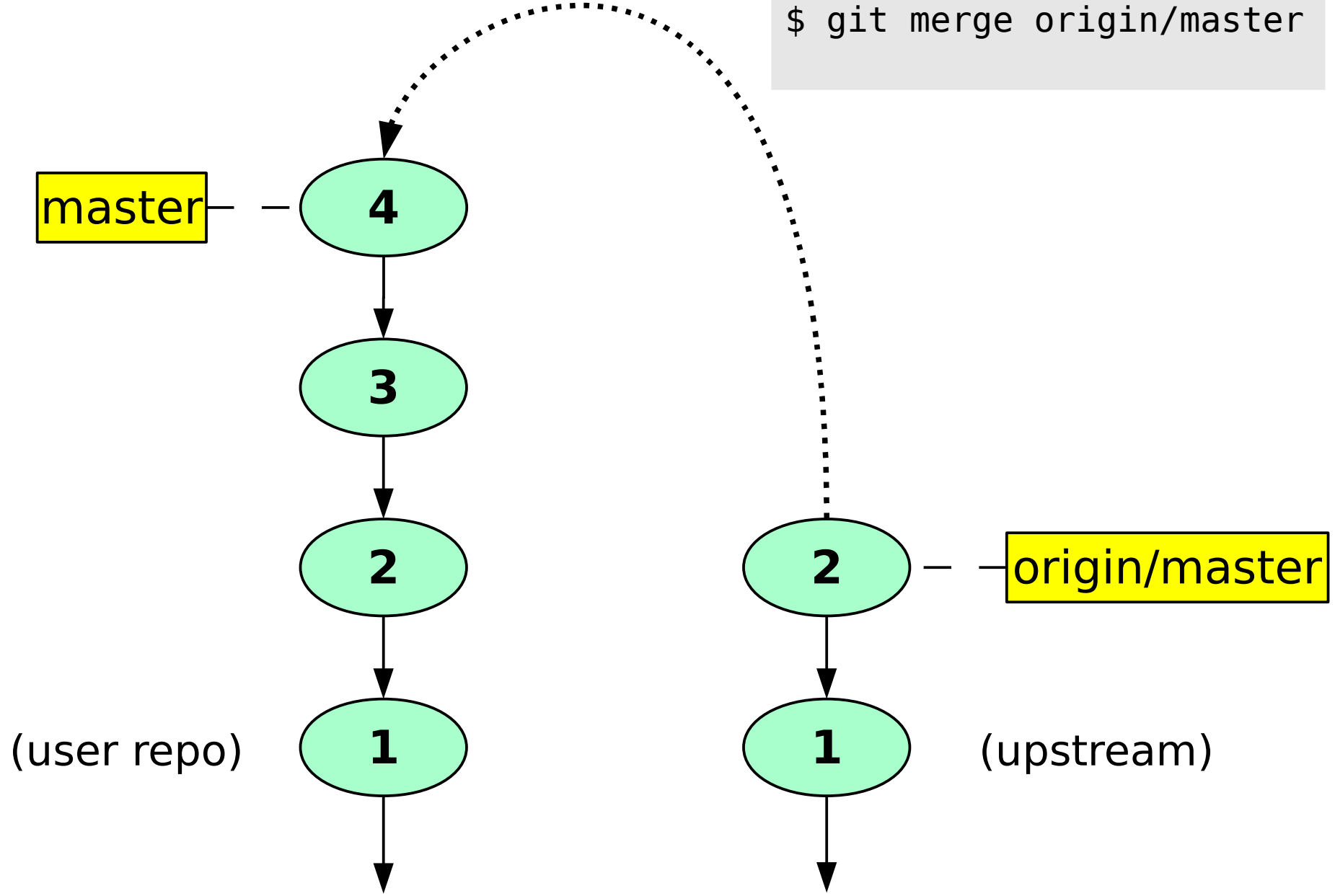
Fast-forward merge

```
$ git merge origin/master
Updating 73e1f20..3cf97be
Fast-forward
```



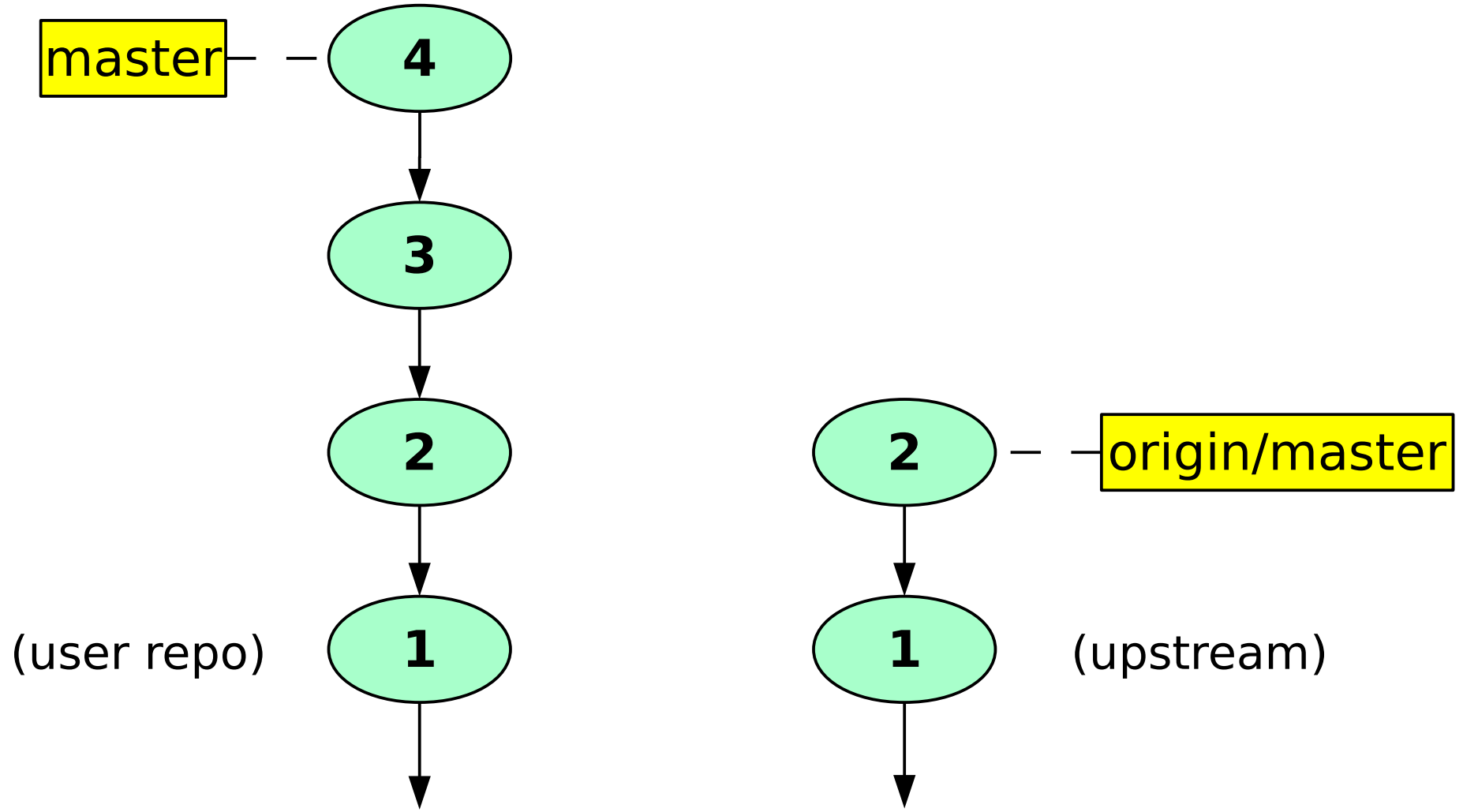
"Already up-to-date" merge

```
$ git merge origin/master
```



"Already up-to-date" merge

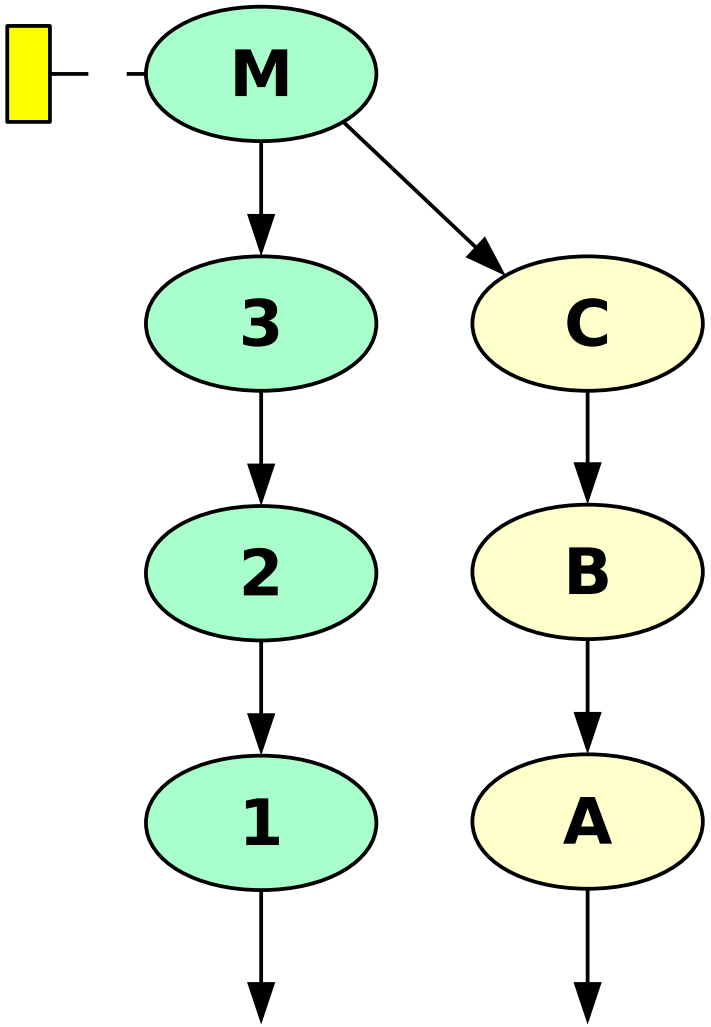
```
$ git merge origin/master
Already up-to-date.
```



(true) Merge

Merge

- Commit with 2 or more parents



Merge

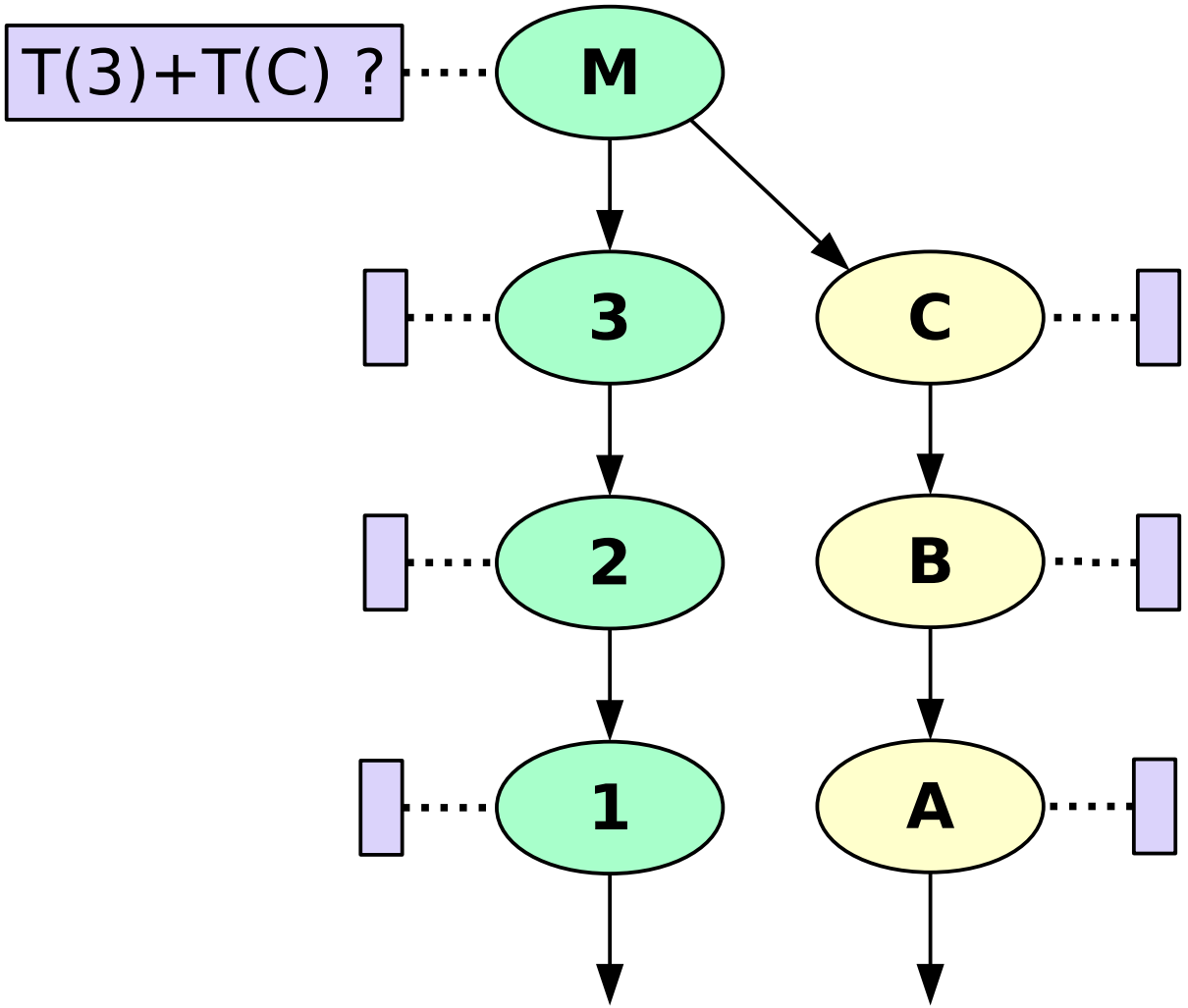
- Commit with 2 or more parents

```
git show --format=raw 7bb8082
```

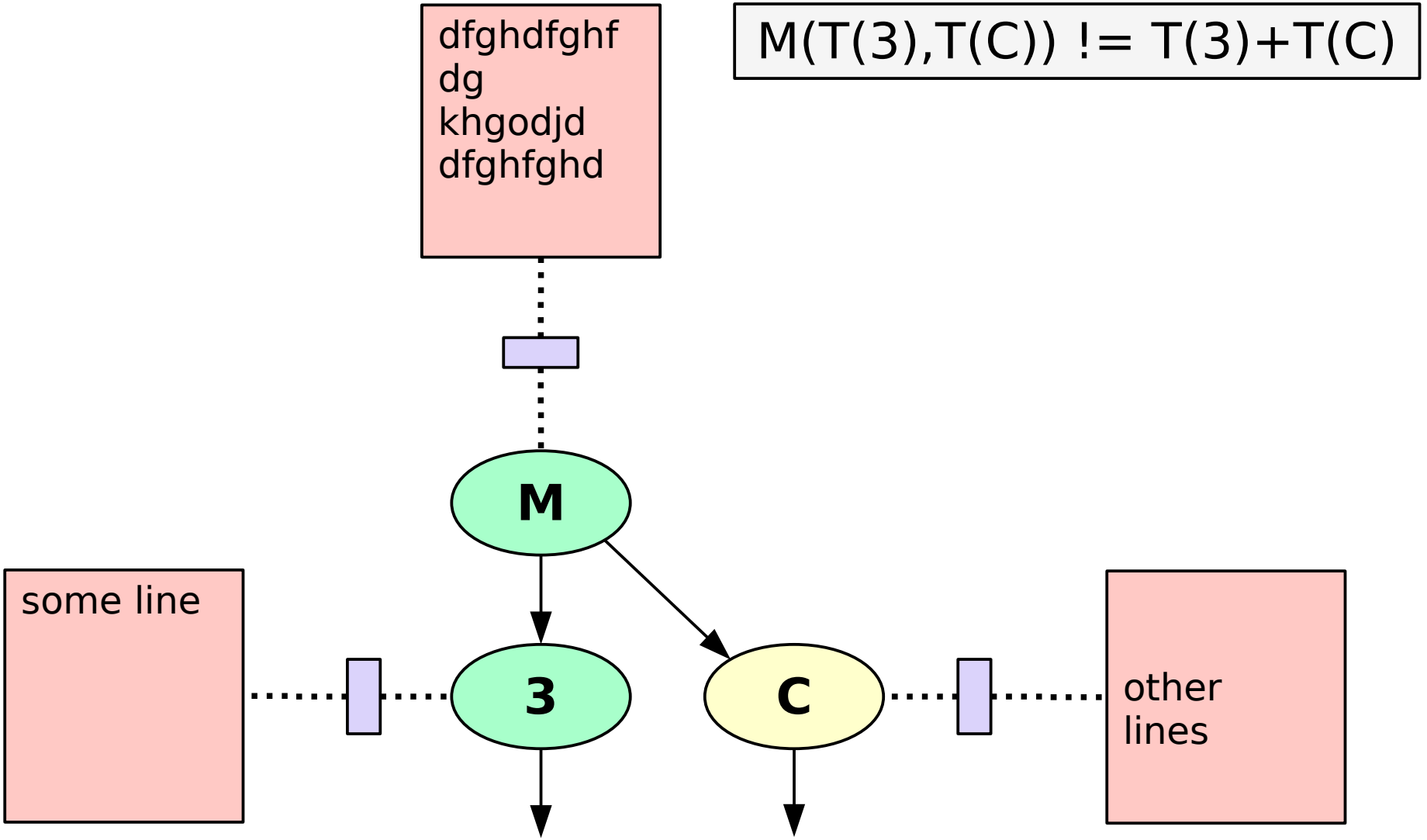
```
commit 7bb8082d7f54325a8dda531ebc4da399d7a90bf5
tree 0248f5d932ac3e979b363cab50df39107fbcd41d
parent 62458e4b11f52c7b96b7b19cf2e51047c5f8bb09
parent 94fcac4d06273976531c463a161f3be0a900de30
author Karel Srot <ksrot@redhat.com> 1330956910 +0100
committer Karel Srot <ksrot@redhat.com> 1330956910 +0100

    Merge remote branch 'origin'
```

Merge

- Commit identifies state



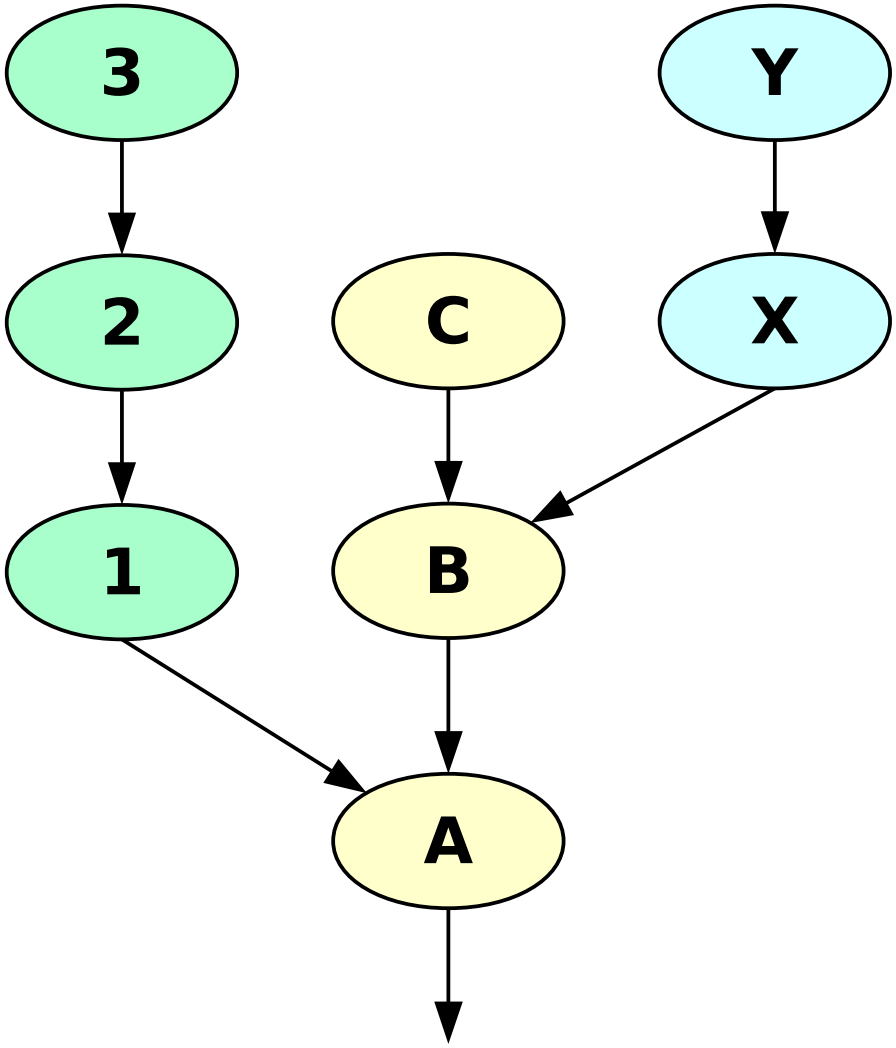
Merge


dfghdfghf
dg
khgodjd
dfghfghd
M(T(3),T(C)) != T(3)+T(C)
M
some line
3
C
other
lines

Common ancestor

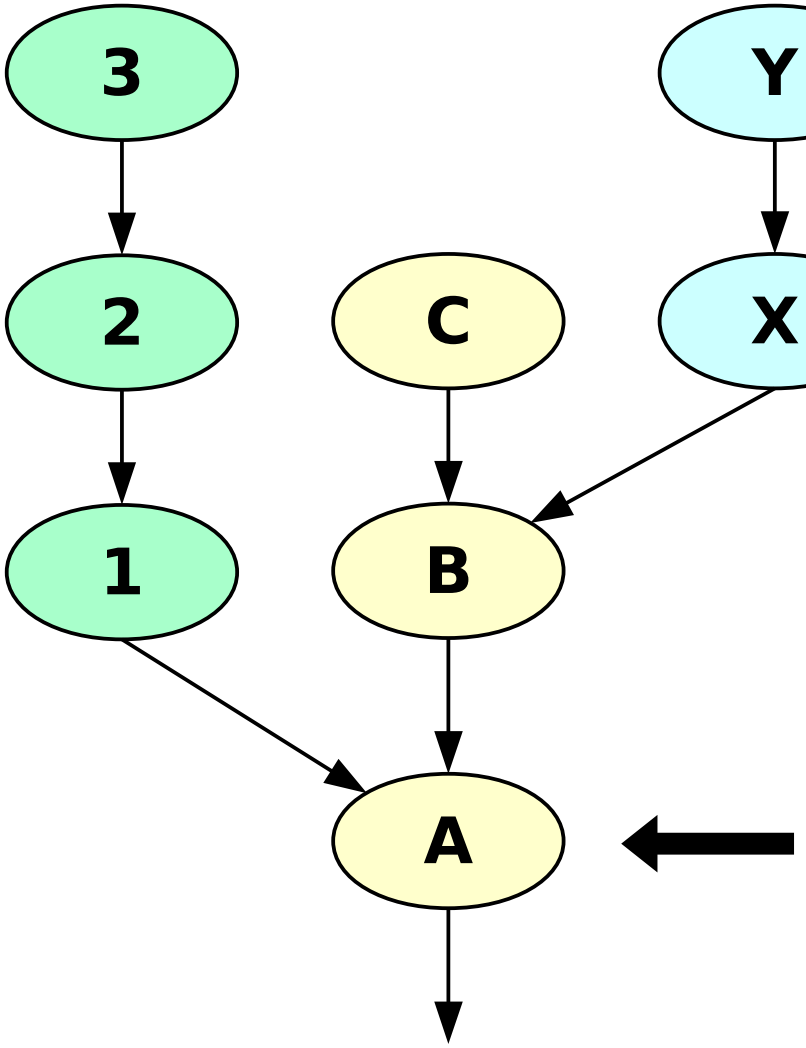
Common ancestor

- Shared dependency base for >=2 branches

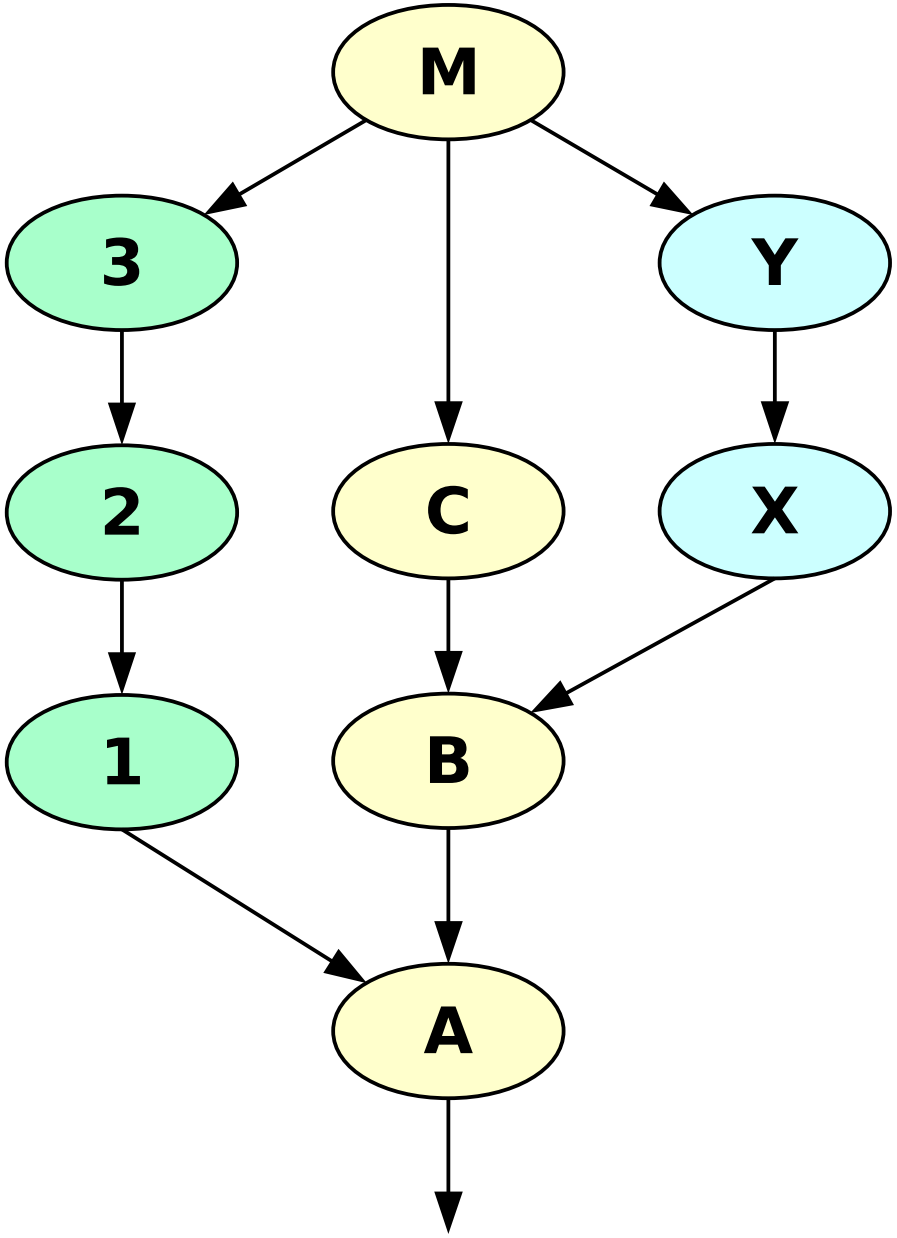


Common ancestor

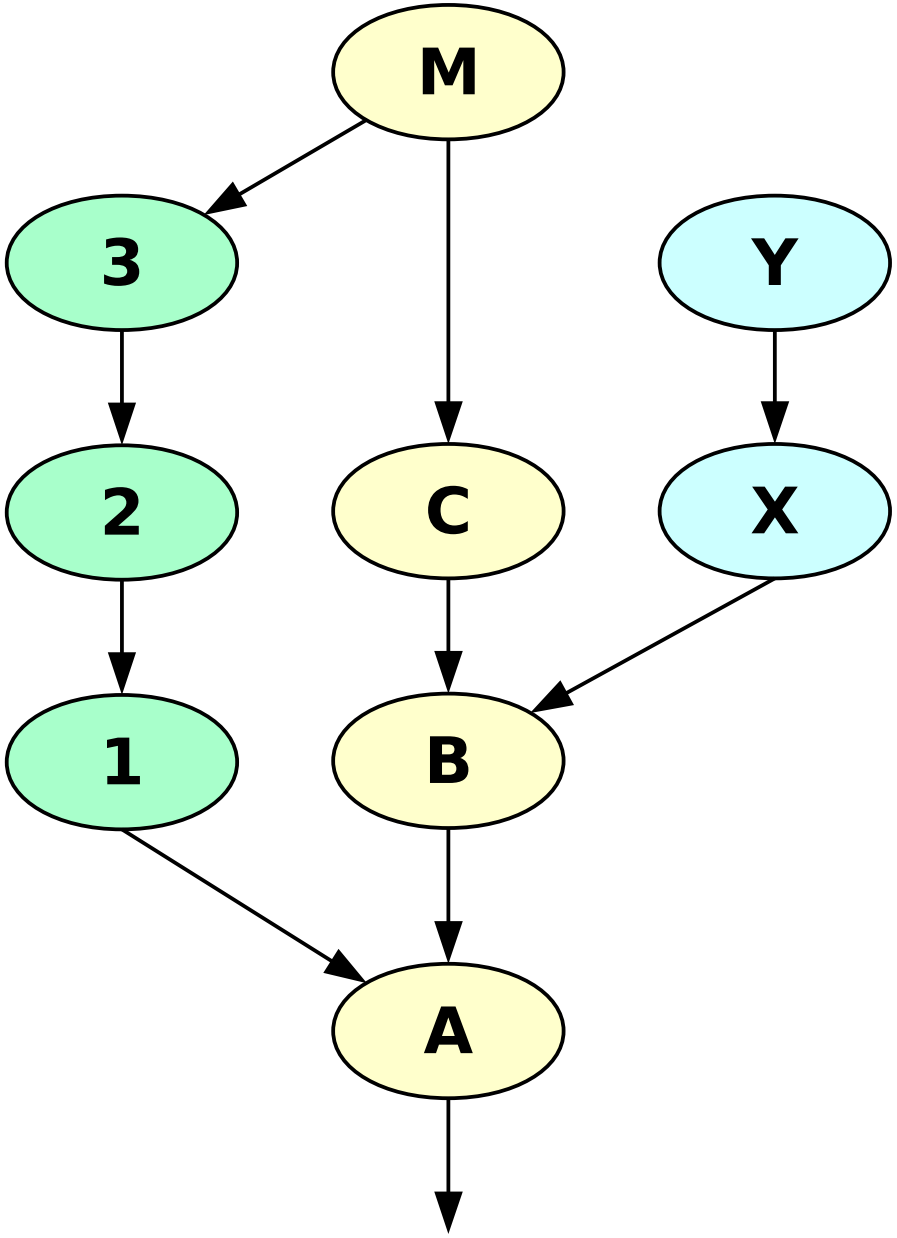
- Shared dependency base for >=2 branches



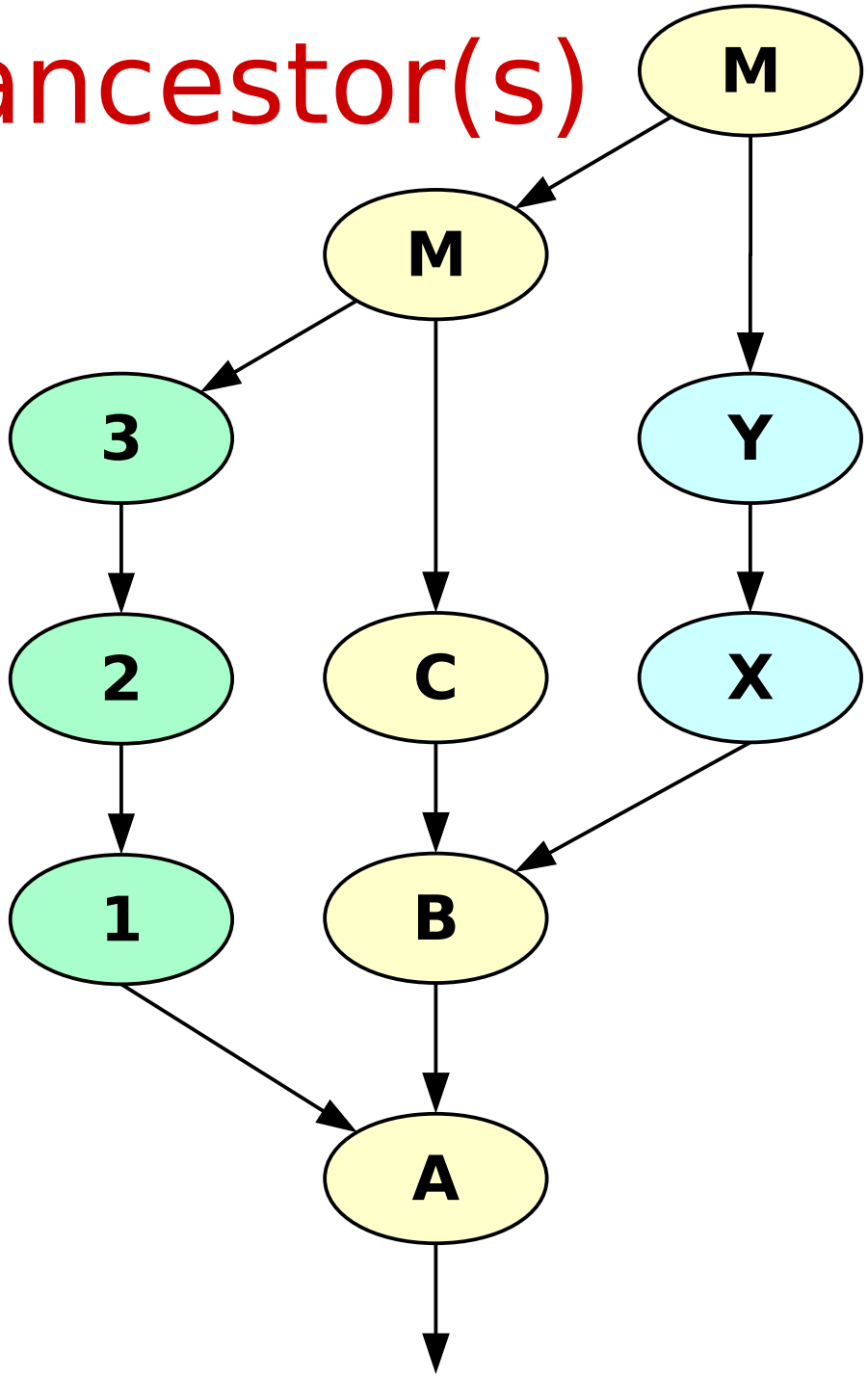
Common ancestor(s)



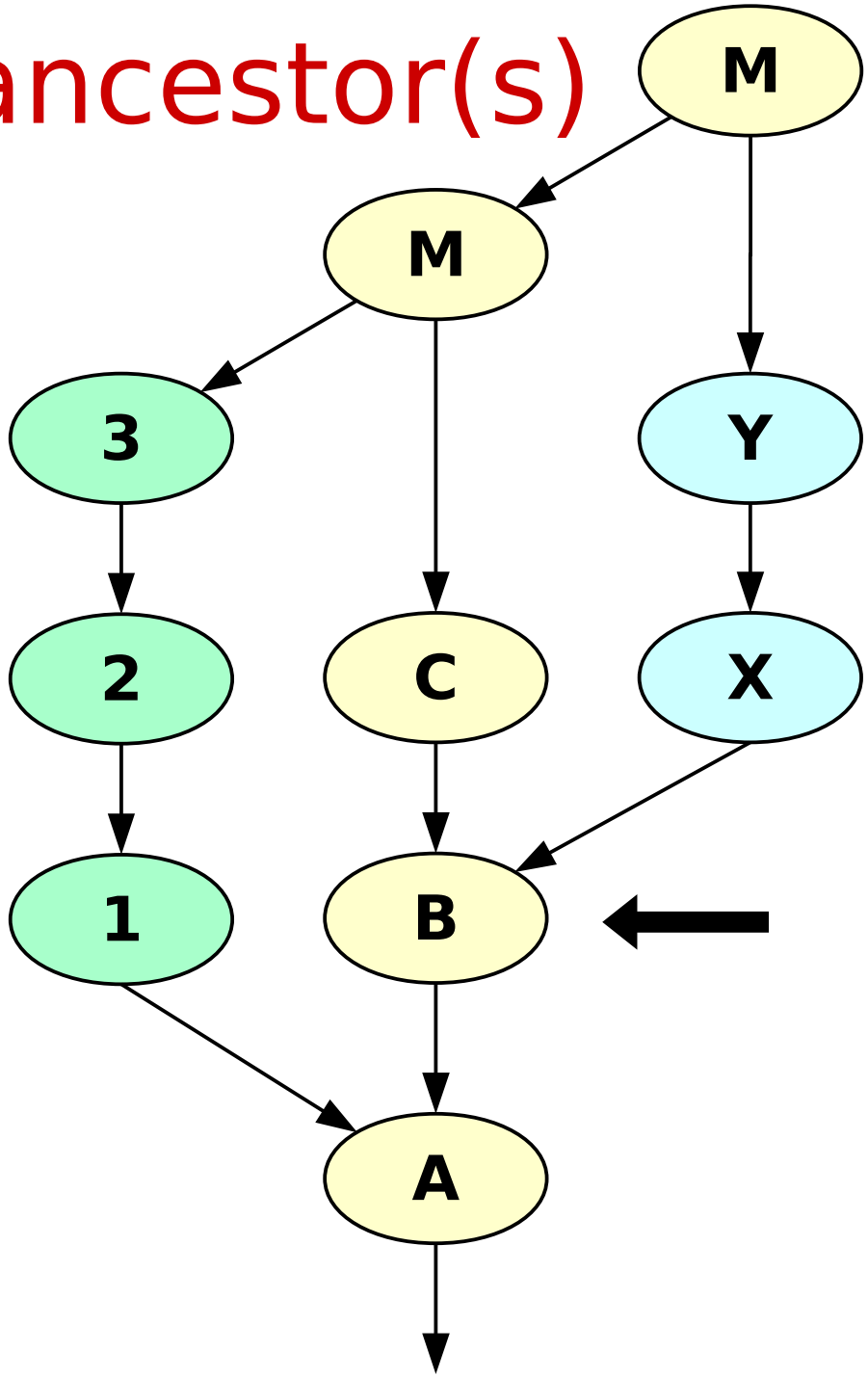
Common ancestor(s)



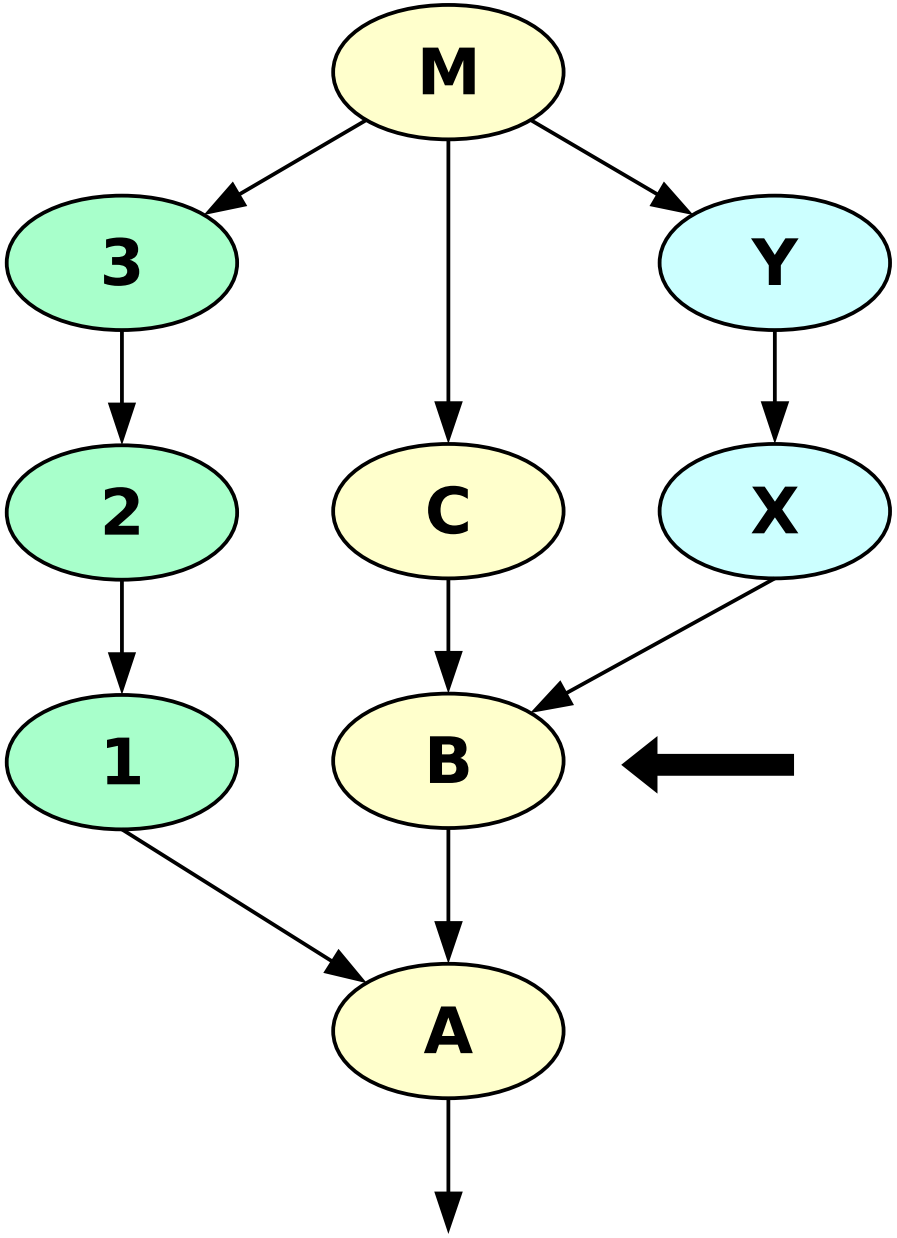
Common ancestor(s)



Common ancestor(s)



Common ancestor(s)



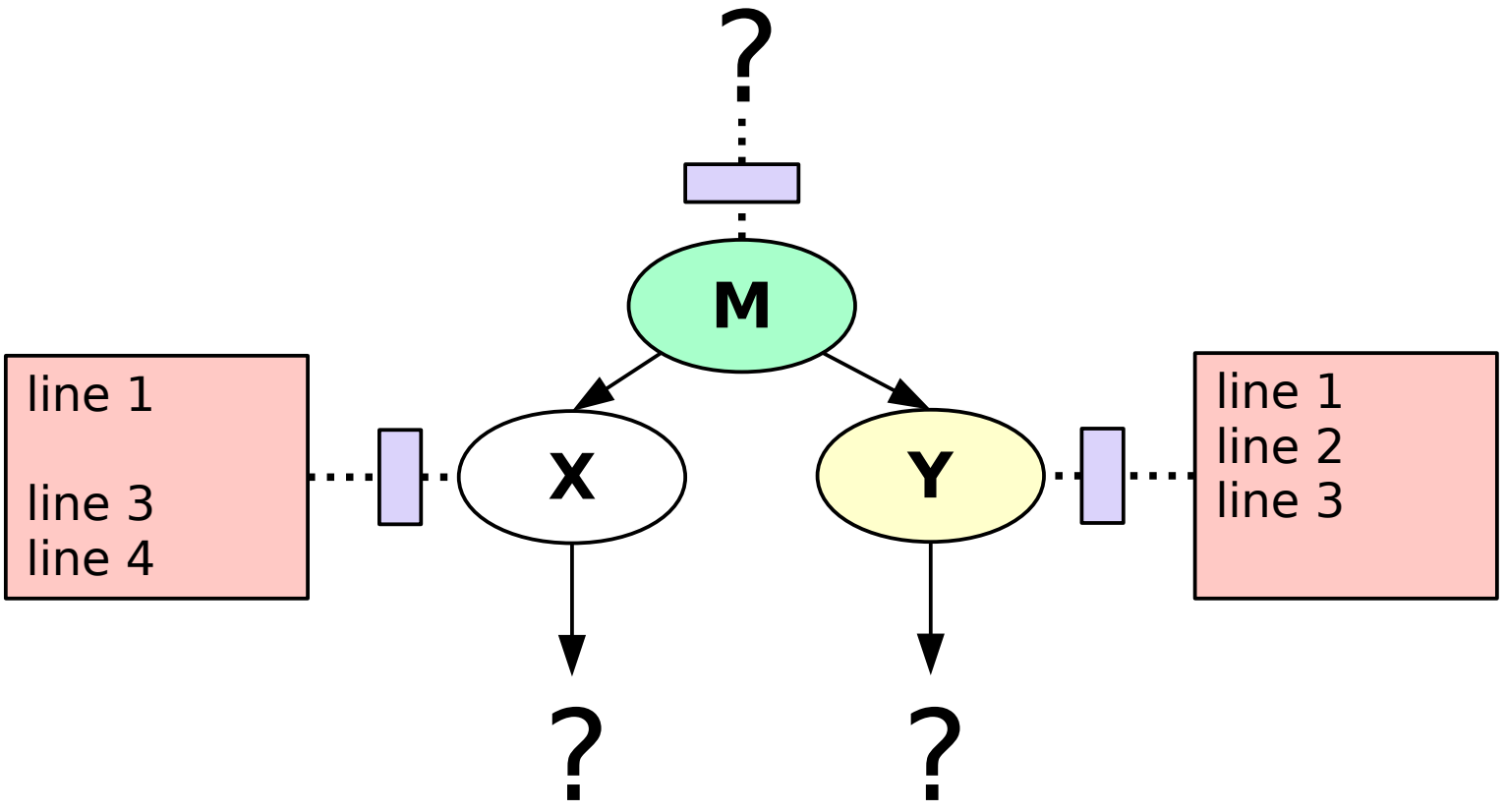
Merge strategies

Merge strategies

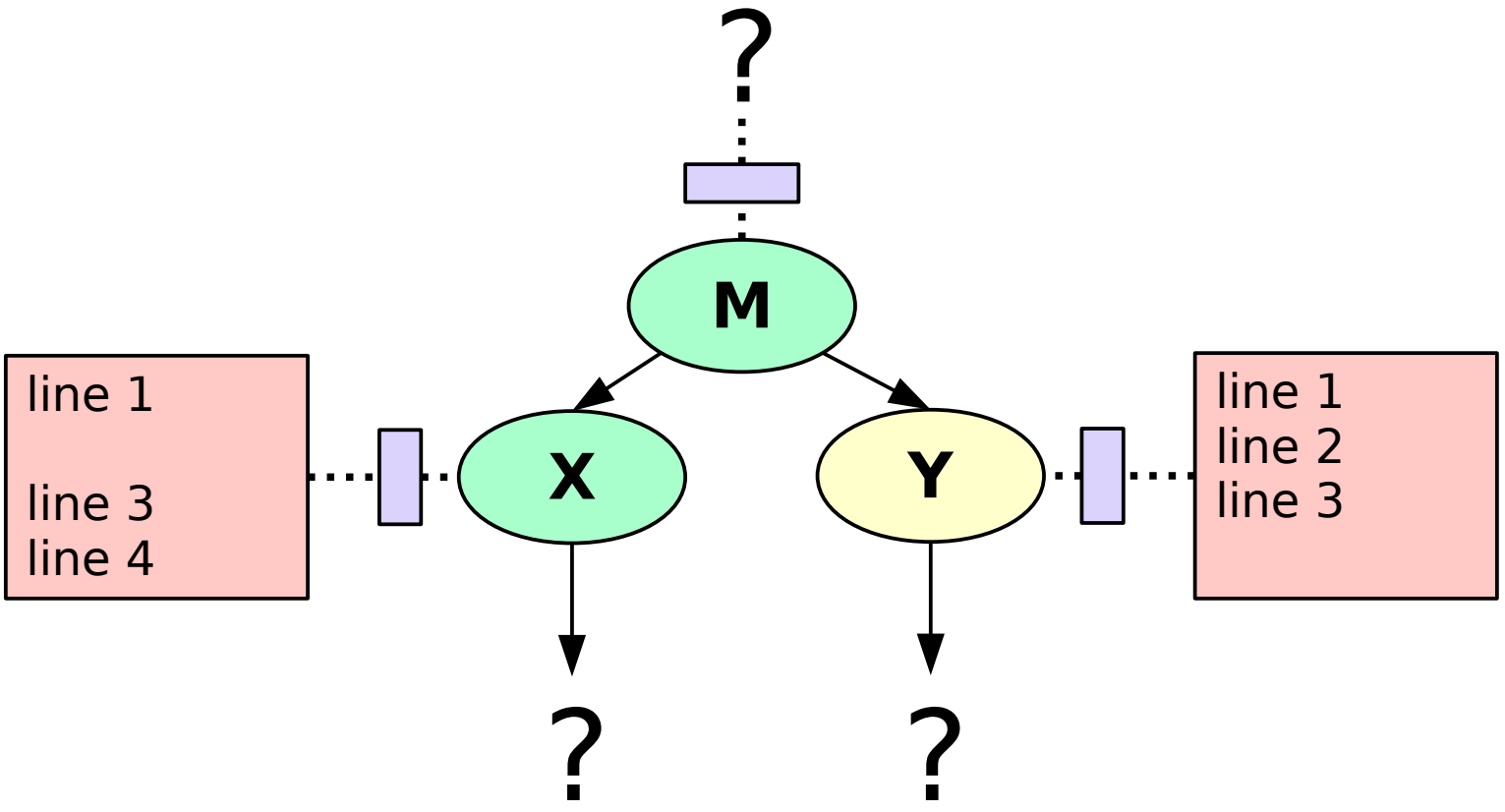
- Resolve
- Recursive
  - Only 2 heads, manual resolution (conflicts)
  - Parametrization
    - ours
    - theirs
    - ...
- Octopus
  - Used for merging many (>2) heads
  - No manual resolution
- Ours

3-way merge

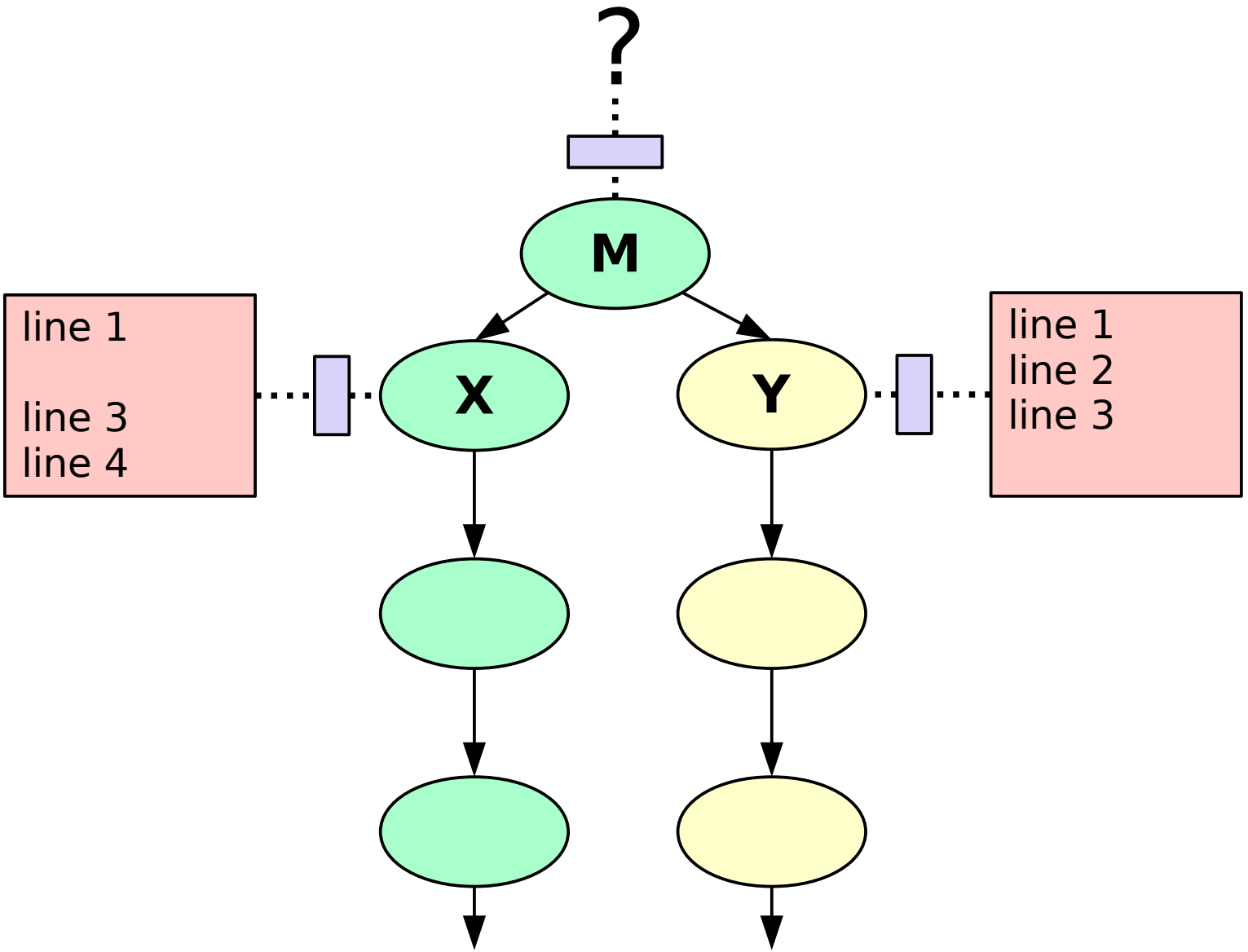
3-way merge



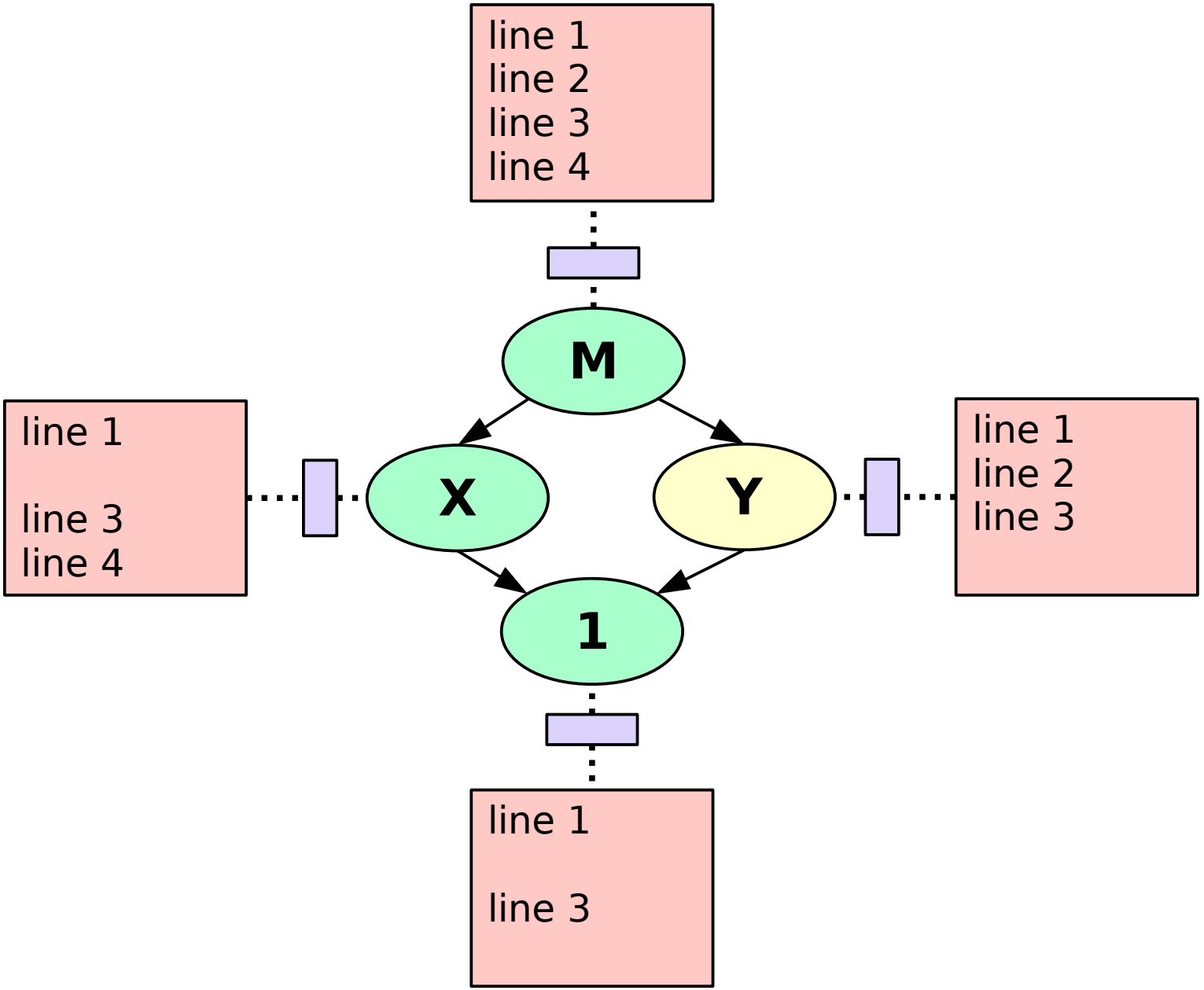
3-way merge



3-way merge

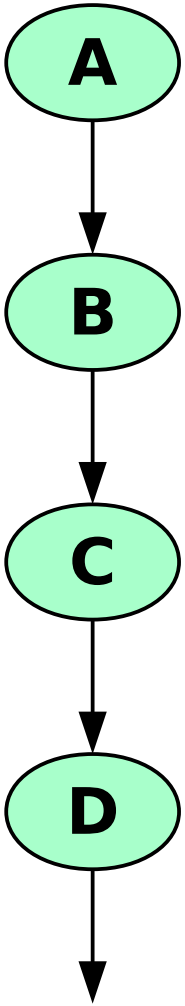


3-way merge


line 1
line 2
line 3
line 4
M
line 1
line 3
line 4
X
Y
line 1
line 2
line 3
1
line 1
line 3

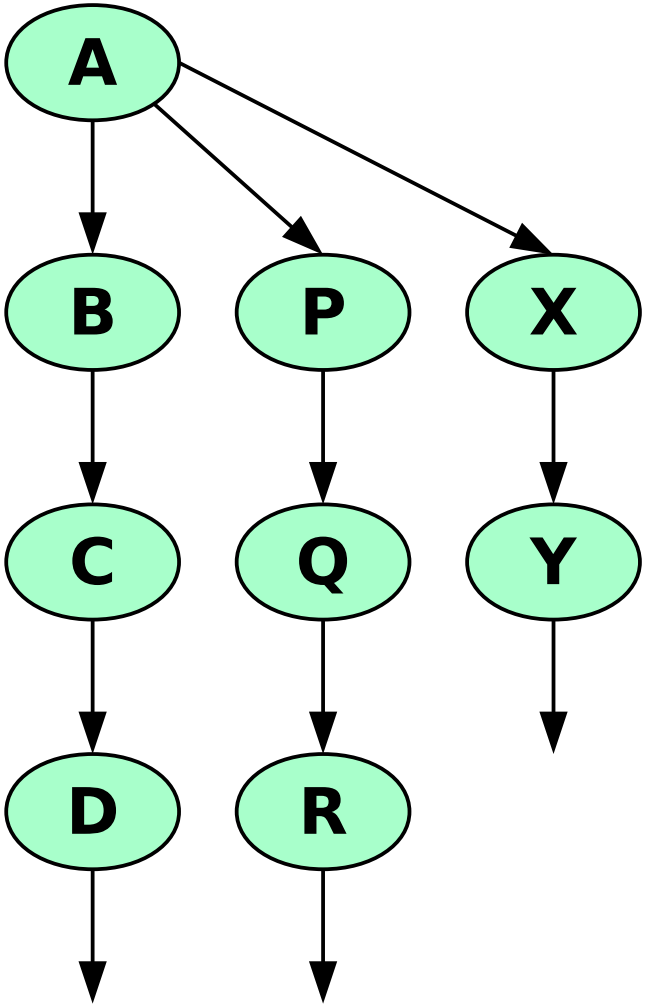
Revision specification

Revision specification




```
A =  A~0
B =  B~0 =  A~1 =  A~
C =  C~0 =  B~1 =  B~ =  A~2
D =  A~3
```

Revision specification

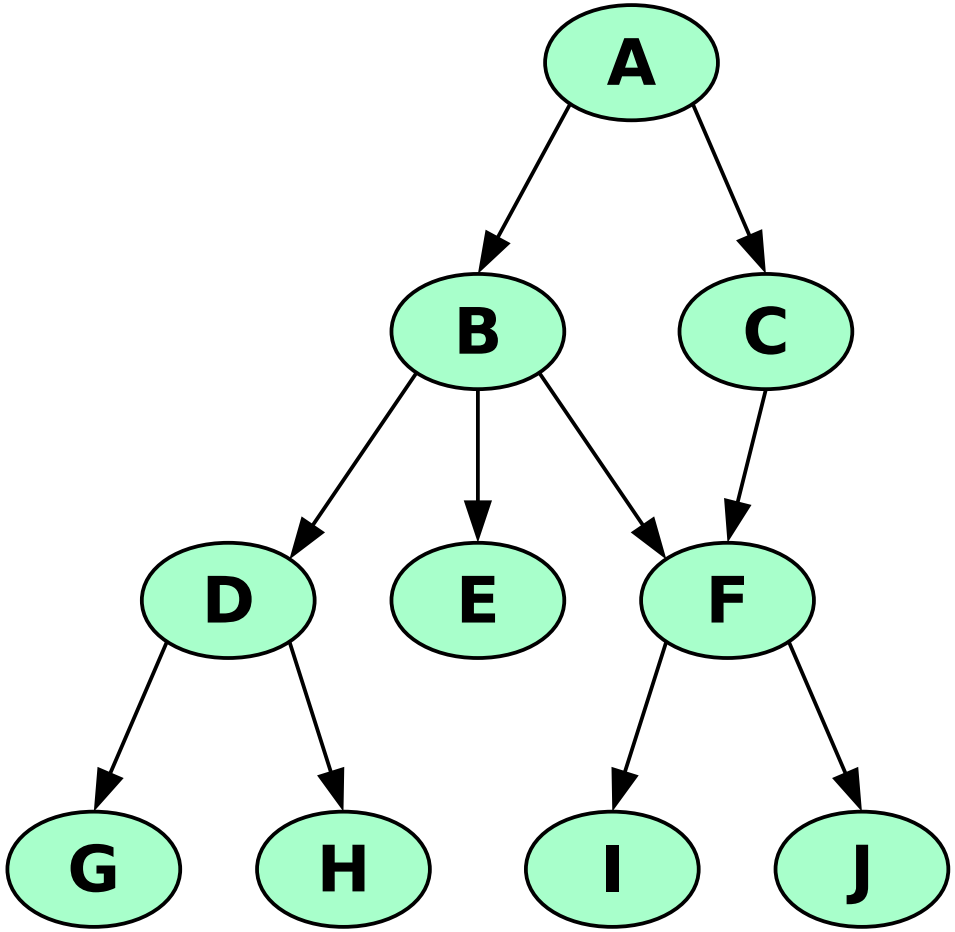



```
A =          = A^0      = A~0
B = A^       = A^1      = A~1
C = A^^      = A^1^1    = A~2
D = A^^^     = A^1^1^1  = A~3

P = A^2
Q = A^2^     = A^2^1
R = A^2^^    = A^2^1^1  = A^2~2

X = A^3
Y = A^3^     = A^3^1    = A^3~
```

Revision specification

(git-rev-parse)

```
man 7 gitrevisions
```




```
A =       = A^0
B = A^    = A^1     = A~1
C = A^2   = A^2
D = A^^   = A^1^1   = A~2
E = B^2   = A^^2
F = B^3   = A^^3
G = A^^^  = A^1^1^1 = A~3
H = D^2   = B^^2    = A^^^2  = A~2^2
I = F^    = B^3^    = A^^3^
J = F^2   = B^3^2   = A^^3^2
```

git-merge

git-merge

- git merge [options] <otherbranch>
  - --no-ff
  - --squash
  - --no-commit
  - ...
- git merge --abort

Summary

Summary

- Degenerate merges
  - Fast-forward
  - Already up-to-date
- (true) Merge
  - Commit with 2 or more parents
    - has one tree, like a regular commit, = merge result
- Common ancestor
- Merge strategies
- 3-way merge
- Revision specification
- git-merge

Presentation: Questions?

Workshop >>

use workshop-05 repository/branch for this workshop

Workshop

goal: **merge {29283,25454,1267}.txt** to master, using git-merge
BONUS: don't use commit hashes

1) **See changes**
   - **git log** ; **git log --oneline --graph --decorate final**
   - **gitk final** ; **tig final** ; **git ls-tree final**
2) **Find out which commits added requested txt files**
   - ie. **git log --oneline --abbrev final -- 29283.txt**
3) **Perform an octopus merge of those commits**
   - **git merge commithash1 commithash2 commithash3**
4) **See result**
   - **git log --oneline --graph --decorate** ; **ls**

- **BONUS part**
  - use hashes from (2) to traverse graph from (1) and apply revision specification knowledge to translate commit{1,2,3} hashes into revspecs starting from the "final" tag
  - example step (3): **git merge final^2~3^4 final^5^^ final^6~7**
    (visual example! wrong revspecs!)
  - hints: **man 7 gitrevisions** ; multiple solutions possible

The End

Thanks for listening

Links

- Git-scm.com - merging, intro to conflicts

```
http://git-scm.com/book/en/Git-Branching-Basic-Branching-and-Merging
```

- Manpage - degenerate merges, true merge, strategies, ...

```
man 1 git-merge
```