mgr - Bug #37940

upmap balancer won't refill underfull osds if zero overfull found

01/16/2019 01:56 PM - Dan van der Ster

Status:	Resolved	**Start date:**	01/16/2019
Priority:	Normal	**Due date:**	
Assignee:	xie xingguo	**% Done:**	0%
Category:	balancer module	**Estimated time:**	0.00 hour
Target version:			
Source:	Community (dev)	**Reviewed:**	
Tags:		**Affected Versions:**	
Backport:	mimic, luminous	**ceph-qa-suite:**	
Regression:	No	**Pull request ID:**	
Severity:	3 - minor		

Description

The following was seen on v12.2.10.

One pool has been upmap balanced for awhile, so there are now zero overfull osds. But there is a new osd (557) which gets stuck severely underfull forever.
Here is the relevant log in calc_pg_upmaps:

```
2019-01-16 14:17:49.103173 7f203d36d700 20  osd.553     pgs 85  target 85.5565  deviation -0.55645
8
2019-01-16 14:17:49.103176 7f203d36d700 20  osd.554     pgs 58  target 57.0376  deviation 0.962406
2019-01-16 14:17:49.103178 7f203d36d700 20  osd.555     pgs 57  target 57.0376  deviation -0.03759
38
2019-01-16 14:17:49.103180 7f203d36d700 20  osd.556     pgs 58  target 57.0376  deviation 0.962406
2019-01-16 14:17:49.103183 7f203d36d700 20  osd.557     pgs 24  target 57.0376  deviation -33.0376
2019-01-16 14:17:49.103185 7f203d36d700 20  osd.558     pgs 58  target 57.0376  deviation 0.962406
2019-01-16 14:17:49.103188 7f203d36d700 20  osd.559     pgs 58  target 57.0376  deviation 0.962406
2019-01-16 14:17:49.103190 7f203d36d700 20  osd.560     pgs 58  target 57.0376  deviation 0.962406
2019-01-16 14:17:49.103192 7f203d36d700 20  osd.561     pgs 58  target 57.0376  deviation 0.962406
2019-01-16 14:17:49.103194 7f203d36d700 20  osd.562     pgs 58  target 57.0376  deviation 0.962406
2019-01-16 14:17:49.103197 7f203d36d700 20  osd.563     pgs 57  target 57.0376  deviation -0.03759
38
2019-01-16 14:17:49.103199 7f203d36d700 20  osd.564     pgs 57  target 57.0376  deviation -0.03759
38
2019-01-16 14:17:49.103201 7f203d36d700 20  osd.565     pgs 57  target 57.0376  deviation -0.03759
38
2019-01-16 14:17:49.103203 7f203d36d700 20  osd.566     pgs 57  target 57.0376  deviation -0.03759
38
2019-01-16 14:17:49.103206 7f203d36d700 20  osd.567     pgs 56  target 57.0376  deviation -1.03759
2019-01-16 14:17:49.103208 7f203d36d700 20  osd.568     pgs 57  target 57.0376  deviation -0.03759
38
2019-01-16 14:17:49.103217 7f203d36d700 20  osd.569     pgs 57  target 57.0376  deviation -0.03759
38
2019-01-16 14:17:49.103223 7f203d36d700 10  total_deviation 144.336 overfull  underfull [557,466,4
69,471,478,483,485,542,567]
2019-01-16 14:17:49.104091 7f203d36d700 10  start deviation 144.336
2019-01-16 14:17:49.104096 7f203d36d700 10  end deviation 144.336
```

In this case, we have no overfull osds, one underfull osd 557 with a large negation deviation. (The handful of other underfull osds have deviation just under minus 1).

But because of this break, that single underfull osd is never re-filled:

```
@@ -4088,8 +4088,8 @@ int OSDMap::calc_pg_upmaps(
    if (overfull.empty() || underfull.empty())
      break;
```

One way to fix this would be to populate overfull more aggressively:

```
diff --git a/src/osd/OSDMap.cc b/src/osd/OSDMap.cc
index 2bb8beb94e..51bc4e7bdf 100644
--- a/src/osd/OSDMap.cc
+++ b/src/osd/OSDMap.cc
@@ -4067,7 +4067,7 @@ int OSDMap::calc_pg_upmaps(
                     << dendl;
       osd_deviation[i.first] = deviation;
       deviation_osd.insert(make_pair(deviation, i.first));
-      if (deviation >= 1.0)
+      if (deviation >= 0.5)  // magic number, maybe 0.1 is better, maybe a configurable
        overfull.insert(i.first);
       total_deviation += abs(deviation);
     }
```

This way, the balancing would continue as long as there are underfull osds.

I can imagine a similar scenario with few outlier overfull and zero underfull osds, but I haven't seen that in the wild yet.

Thoughts?

Related issues:

Copied to mgr - Backport #38036: mimic: upmap balancer won't refill underfull...	**Resolved**
Copied to mgr - Backport #38037: luminous: upmap balancer won't refill underf...	**Resolved**

History

#1 - 01/19/2019 08:42 AM - xie xingguo

- Assignee set to xie xingguo

#2 - 01/21/2019 04:52 PM - Dan van der Ster

Today I saw the opposite case: calc_pg_upmaps cannot find any upmaps to move PGs out of the most overfull osd.

I guess that this is because for each pg it tries to move, none of the up osds for that PG are in the underfull list **and** in a unique crush bucket.

```
2019-01-21 15:15:02.474148 7fc1ecc33700 10  total_deviation 1608.3 overfull 23,34,41,54,62,73,81,94,108,117,15
4,167,181,205,207,229,239,241,282,304,357,365,374,386,465,510,511,512,513,514,515,516,517,518,5
19,520,521,522,523,524,525,526,528,529,533,534,535,536,537,538,1137,1139,1142,1144,1151,1156,1159,1164,1168,11
74,1179,1182,1186,1189,1204,1205,1210,1214,1215,1221,1222,1223,1224,1225,1226,1228,1231,1232,12
35,1237,1239,1240,1241,1242,1244,1245,1246,1247,1248,1249,1250,1251,1253 underfull [1121,80,192,183,468,491,15
2,155,184,406,1043,1045,1057,1062,1067,1081,1083,1089,1094,1099,1102,78,232,246,280,347,400,482
,464,1046,1054,1078,1087,1127,1298,10,13,17,48,87,121,136,180,273,371,383,500,8,409,1028,1035,1036,1050,1056,1
068,1069,1077,1079,1092,1096,1097,1103,1271,1278,1281,1296,1297,1328,1334,4,46,63,93,116,135,14
0,144,168,228,237,276,288,294,300,327,337,353,392,128,452,463,1022,1024,1031,1034,1037,1041,1052,1059,1060,106
4,1074,1076,1082,1255,1266,1269,1283,1286,1290,1310,1323,1335,16,26,35,43,82,85,153,161,172,173,218,224,242,25
9,266,284,295,303,308,318,326,334,335,366,369,419,474,1038,1044,1058,1070,1075,1080,1084,1091,1093,1107,1108,1
111,1118,1128,1254,1280,1282,1306,1331,1,22,27,33,55,61,75,90,95,114,119,127,164,175,190,194,212,217,255,258,2
71,310,311,325,363,375,404,470,487,493,506,349,1021,1029,1032,1033,1039,1042,1047,1049,1051,1063,1071,1086,109
8,1105,1109,1112,1113,1115,1120,1122,1124,1125,1160,1257,1258,1264,1267,1272,1274,1277,1284,1292,1301,1302,131
4,1327]
2019-01-21 15:15:02.474204 7fc1ecc33700 10  osd.1239 move 25
2019-01-21 15:15:02.474342 7fc1ecc33700 10   trying 68.7e
2019-01-21 15:15:02.475060 7fc1ecc33700 10   trying 68.9d
2019-01-21 15:15:02.475707 7fc1ecc33700 10   trying 68.e2
2019-01-21 15:15:02.476336 7fc1ecc33700 10   trying 68.4fb
2019-01-21 15:15:02.476971 7fc1ecc33700 10   trying 68.b65
2019-01-21 15:15:02.477596 7fc1ecc33700 10   trying 68.df3
2019-01-21 15:15:02.478223 7fc1ecc33700 10   trying 68.e07
```

```
2019-01-21 15:15:02.478854 7fc1ecc33700 10   trying 68.e95
2019-01-21 15:15:02.479505 7fc1ecc33700 10   trying 68.1060
2019-01-21 15:15:02.480145 7fc1ecc33700 10   trying 68.1085
2019-01-21 15:15:02.480806 7fc1ecc33700 10   trying 68.1107
2019-01-21 15:15:02.481452 7fc1ecc33700 10   trying 68.130b
2019-01-21 15:15:02.482079 7fc1ecc33700 10   trying 68.13a9
2019-01-21 15:15:02.482727 7fc1ecc33700 10   trying 68.1581
2019-01-21 15:15:02.483352 7fc1ecc33700 10   trying 68.1590
2019-01-21 15:15:02.483974 7fc1ecc33700 10   trying 68.160f
2019-01-21 15:15:02.484612 7fc1ecc33700 10   trying 68.174f
2019-01-21 15:15:02.485276 7fc1ecc33700 10   trying 68.18c2
2019-01-21 15:15:02.485903 7fc1ecc33700 10   trying 68.1a3b
2019-01-21 15:15:02.486525 7fc1ecc33700 10   trying 68.1b4a
2019-01-21 15:15:02.487152 7fc1ecc33700 10   trying 68.1f19
2019-01-21 15:15:02.487797 7fc1ecc33700 10   trying 68.207d
2019-01-21 15:15:02.488428 7fc1ecc33700 10   trying 68.2123
2019-01-21 15:15:02.489047 7fc1ecc33700 10   trying 68.216e
2019-01-21 15:15:02.489675 7fc1ecc33700 10   trying 68.221c
2019-01-21 15:15:02.490300 7fc1ecc33700 10   trying 68.22cb
2019-01-21 15:15:02.490917 7fc1ecc33700 10   trying 68.23cf
2019-01-21 15:15:02.491588 7fc1ecc33700 10   trying 68.2483
2019-01-21 15:15:02.492225 7fc1ecc33700 10   trying 68.25c7
2019-01-21 15:15:02.492856 7fc1ecc33700 10   trying 68.268b
2019-01-21 15:15:02.493479 7fc1ecc33700 10   trying 68.26c3
2019-01-21 15:15:02.494100 7fc1ecc33700 10   trying 68.275d
2019-01-21 15:15:02.494739 7fc1ecc33700 10   trying 68.27c4
2019-01-21 15:15:02.495363 7fc1ecc33700 10   trying 68.28ff
2019-01-21 15:15:02.495988 7fc1ecc33700 10   trying 68.294a
2019-01-21 15:15:02.496611 7fc1ecc33700 10   trying 68.2b19
2019-01-21 15:15:02.497237 7fc1ecc33700 10   trying 68.2bcf
2019-01-21 15:15:02.497865 7fc1ecc33700 10   trying 68.2c00
2019-01-21 15:15:02.498494 7fc1ecc33700 10   trying 68.2c12
2019-01-21 15:15:02.499113 7fc1ecc33700 10   trying 68.2c16
2019-01-21 15:15:02.499751 7fc1ecc33700 10   trying 68.2e36
2019-01-21 15:15:02.500383 7fc1ecc33700 10   trying 68.2f2a
2019-01-21 15:15:02.501042 7fc1ecc33700 10   trying 68.3031
2019-01-21 15:15:02.501687 7fc1ecc33700 10   trying 68.30de
2019-01-21 15:15:02.502341 7fc1ecc33700 10   trying 68.320b
2019-01-21 15:15:02.502976 7fc1ecc33700 10   trying 68.3377
2019-01-21 15:15:02.503610 7fc1ecc33700 10   trying 68.33fa
2019-01-21 15:15:02.504224 7fc1ecc33700 10   trying 68.341b
2019-01-21 15:15:02.504867 7fc1ecc33700 10   trying 68.346b
2019-01-21 15:15:02.505520 7fc1ecc33700 10   trying 68.3692
2019-01-21 15:15:02.506162 7fc1ecc33700 10   trying 68.3708
2019-01-21 15:15:02.506794 7fc1ecc33700 10   trying 68.3873
2019-01-21 15:15:02.507427 7fc1ecc33700 10   trying 68.38d3
2019-01-21 15:15:02.508047 7fc1ecc33700 10   trying 68.3a8f
2019-01-21 15:15:02.508690 7fc1ecc33700 10   trying 68.3acf
2019-01-21 15:15:02.509317 7fc1ecc33700 10   trying 68.3c79
2019-01-21 15:15:02.509940 7fc1ecc33700 10   trying 68.3e24
2019-01-21 15:15:02.510571 7fc1ecc33700 10  osd.1226 move 23
...
```

So perhaps here it will also be better to more aggressively fill the underfull osds list.

#3 - 01/24/2019 12:59 AM - xie xingguo

https://github.com/ceph/ceph/pull/26039

#4 - 01/24/2019 01:00 AM - xie xingguo

- Status changed from New to Feedback

#5 - 01/24/2019 02:37 PM - Dan van der Ster

Thanks, this looks great!
Do you already have a luminous branch with this backported? I can eventually build and try it here, but the cherry-pick is a non-trivial merge.

#6 - 01/24/2019 03:21 PM - Nathan Cutler

- Status changed from Feedback to Pending Backport

- Backport set to mimic, luminous

#7 - 01/24/2019 03:22 PM - Nathan Cutler

- Copied to Backport #38036: mimic: upmap balancer won't refill underfull osds if zero overfull found added

#8 - 01/24/2019 03:22 PM - Nathan Cutler

- Copied to Backport #38037: luminous: upmap balancer won't refill underfull osds if zero overfull found added

#9 - 01/25/2019 01:57 PM - Dan van der Ster

Thanks Xie! This is helping, but I think there are still some improvements needed.

1) With this code, I see lots of improvements to the general balancing, but unfortunately our most overfull osds are unlucky and not getting balanced, because all of the possible PG upmaps to make are already remapped (or not remappable). For example:

```
2019-01-25 14:28:32.225243 7fc6be64e700 10  osd.1239 move 27
2019-01-25 14:28:32.225387 7fc6be64e700 20   already remapped 68.16
2019-01-25 14:28:32.225390 7fc6be64e700 20   already remapped 68.3a
2019-01-25 14:28:32.225391 7fc6be64e700 20   already remapped 68.5f
2019-01-25 14:28:32.225393 7fc6be64e700 20   already remapped 68.64
2019-01-25 14:28:32.225394 7fc6be64e700 10   trying 68.7e
2019-01-25 14:28:32.226151 7fc6be64e700 10   trying 68.9d
2019-01-25 14:28:32.226883 7fc6be64e700 10   trying 68.e2
2019-01-25 14:28:32.227656 7fc6be64e700 20   already remapped 68.20a
2019-01-25 14:28:32.227668 7fc6be64e700 20   already remapped 68.27b
2019-01-25 14:28:32.227669 7fc6be64e700 20   already remapped 68.28c
...
2019-01-25 14:28:32.261401 7fc6be64e700 10   trying 68.3e24
2019-01-25 14:28:32.262133 7fc6be64e700 20   already remapped 68.3e31
2019-01-25 14:28:32.262140 7fc6be64e700 20   already remapped 68.3f12
2019-01-25 14:28:32.262141 7fc6be64e700 20   already remapped 68.3f56
2019-01-25 14:28:32.262142 7fc6be64e700 20   already remapped 68.3f7b
2019-01-25 14:28:32.262143 7fc6be64e700 20   already remapped 68.3ffa
2019-01-25 14:28:32.262145 7fc6be64e700 10  osd.1226 move 23
...
```

Here are some of those existing upmaps which prevent 1239 from moving off some PGs:

```
pg_upmap_items 68.16 [1318,1271]  // now [1239,1036,1150,180,325,1271]
pg_upmap_items 68.3a [1030,80]    // now [1111,1267,1054,1208,1239,80]
pg_upmap_items 68.5f [1158,1133]  // now [1239,1063,1293,824,1133,1108]
```

and here are the osd's involved in those upmaps:

```
ID   CLASS WEIGHT     REWEIGHT SIZE    USE     AVAIL   %USE  VAR  PGS TYPE NAME
1318   hdd    5.45799  1.00000 5.46TiB 4.00TiB 1.46TiB 73.23 1.01 184                 osd.1318
1271   hdd    5.45799  1.00000 5.46TiB 3.85TiB 1.60TiB 70.61 0.98 176                 osd.1271
```

```
1030   hdd    5.45799  1.00000 5.46TiB 4.02TiB 1.43TiB 73.72 1.02 185                     osd.1030
  80   hdd    5.43700  1.00000 5.46TiB 3.98TiB 1.48TiB 72.91 1.01 183                     osd.80
1158   hdd    5.45799  1.00000 5.46TiB 4.02TiB 1.44TiB 73.65 1.02 185                     osd.1158
1133   hdd    5.45799  1.00000 5.46TiB 4.02TiB 1.44TiB 73.68 1.02 185                     osd.1133
```

The code printing "already remapped" is as follows:

```
    if (tmp.have_pg_upmaps(pg)) {
      ldout(cct, 20) << "  already remapped " << pg << dendl;
      continue;
    }
```

Could we add some logic to append a 2nd, 3rd, etc... pair to existing upmaps, rather than just continuing to the next PG?

2) A second optimization: I noticed that we only call rm-pg-upmap-items to remove an existing upmap which remaps to an overfull OSD. We should also handle the underfull case: we can rm-pg-upmap-items if there exist any upmaps which remapped a PG out from an underfull OSD.

Thanks!

#10 - 01/25/2019 08:22 PM - Dan van der Ster

Regarding my point (1) above, I think I've understood out the real issue. In my cluster, we have a 4+2 EC pool and 6 racks of servers. Due to some disks being down, the racks are identical in crush weight but one rack has slightly fewer up/in OSDs than the others.

So basically, all OSDs in this rack are overfull, and due to the constraint that pool size = num racks in my cluster, there are very few, or no underfill OSDs to remap to.

Here's a toy example to demonstrate this case:

3.0 rack a
1.0 OSD.0: 2 pgs
1.0 OSD.1: 2 pgs
1.0 OSD.2: 2 pgs
3.0 rack b
1.0 OSD.3: 2 pgs
1.0 OSD.4: 2 pgs
1.0 OSD.5: 2 pgs
3.0 rack c
1.0 OSD.6: 2 pgs
1.0 OSD.7: 4 pgs
1.0 OSD.8: 0 pgs (is down/out)

Pool has size 3, pg_num 6, replicated across racks.

In that toy example, I think we **want** an upmap to balance osds 6 and 7 with 3 pgs each. But it won't because there are no underfull osds in rack c.

#11 - 03/12/2019 02:20 PM - Nathan Cutler

- Status changed from Pending Backport to Resolved