cregit: identifying contributors of source code

Daniel M German
University of Victoria
dmg@uvic.ca

Kate Stewart
Linux Foundation

Bram Adams
Polytechnique of Montreal
Linux
Linux is not only software...

it is also software engineering
Git promotes Linux practices:
Git promotes Linux practices:

- code reviews
Git promotes Linux practices:

- code reviews
- pull requests
Git promotes Linux practices:

- code reviews
- pull requests
- isolation
Git promotes Linux practices:

- code reviews
- pull requests
- isolation

and github popularizes them
GitHub for non-technical people

FROM COLLABORATIVE CODING TO WEDDING INVITATIONS: GITHUB IS GOING MAINSTREAM

GitHub for the rest of us

Git made it possible for programmers to coordinate distributed work across teams -- now GitHub makes it possible for everyone else.
cregit: increasing provenance information
static struct alloc_state commit_state;

void *alloc_commit_node(void)
{
    static int commit_count;
    struct commit *c = alloc_node(&commit_state, sizeof(struct commit));
    c->object.type = OBJ_COMMIT;
    c->index = commit_count++;
    return c;
}

unsigned int alloc_commit_index(void)
{
    static unsigned int count;
    return count++;
}
IF YOU MUST SQUAWK, BLAME THEM - AND WATCH OUT FOR ME!
<table>
<thead>
<tr>
<th>Patch Title</th>
<th>Author</th>
<th>Date</th>
<th>Lines</th>
<th>Include Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>mmu-notifiers: core</td>
<td>Andrea Arcangeli</td>
<td>28 Jul 2008</td>
<td>29</td>
<td><code>&lt;linux/mmu_notifier.h&gt;</code></td>
</tr>
<tr>
<td>Linux-2.6.12-rc2</td>
<td>Linus Torvalds</td>
<td>16 Apr 2005</td>
<td>30</td>
<td><code>&lt;linux/fs.h&gt;</code></td>
</tr>
<tr>
<td>mm: per-thread vma caching</td>
<td>Davidlohr Bueso</td>
<td>7 Apr 2014</td>
<td>31</td>
<td><code>&lt;linux/mm.h&gt;</code></td>
</tr>
<tr>
<td>[PATCH] namespaces: add nsprp...</td>
<td>Serge P. Hallyn</td>
<td>2 Oct 2006</td>
<td>32</td>
<td><code>&lt;linux/vmcache.h&gt;</code></td>
</tr>
<tr>
<td>[PATCH] move capable() to capa...</td>
<td>Randy.Dunlap</td>
<td>11 Jan 2006</td>
<td>33</td>
<td><code>&lt;linux/nsproxy.h&gt;</code></td>
</tr>
<tr>
<td>Linux-2.6.12-rc2</td>
<td>Linus Torvalds</td>
<td>16 Apr 2005</td>
<td>34</td>
<td><code>&lt;linux/capability.h&gt;</code></td>
</tr>
<tr>
<td>Task Control Groups: add fork()...</td>
<td>Paul Menage</td>
<td>19 Oct 2007</td>
<td>35</td>
<td><code>&lt;linux/cpu.h&gt;</code></td>
</tr>
<tr>
<td>Linux-2.6.12-rc2</td>
<td>Linus Torvalds</td>
<td>16 Apr 2005</td>
<td>36</td>
<td><code>&lt;linux/cgroup.h&gt;</code></td>
</tr>
<tr>
<td>hugetlb: reserve huge pages for...</td>
<td>gormannm</td>
<td>24 Jul 2008</td>
<td>37</td>
<td><code>&lt;linux/security.h&gt;</code></td>
</tr>
<tr>
<td>seccomp: add system call filteri...</td>
<td></td>
<td></td>
<td>38</td>
<td><code>&lt;linux/hugetlb.h&gt;</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td><code>&lt;linux/seccomp.h&gt;</code></td>
</tr>
</tbody>
</table>
“History is written by the winners”
7.6 Git Tools - Rewriting History

Rewriting History

Many times, when working with Git, you may want to revise your commit history for some reason. One of the great things about Git is that it allows you to make decisions at the last possible moment. You can decide what files go into which commits right before you commit with the staging area, you can decide that you didn’t mean to be working on something yet with the stash command, and you can rewrite commits that already happened so they look like they happened in a different way. This can involve changing the order of the commits, changing messages or modifying files in a commit, squashing together or splitting apart commits, or removing commits entirely – all before you share your work with others.
“Archeology is the search for fact, not truth. If it's truth you're interested in, Dr. Tyree's Philosophy class is right down the hall.”

-- Indiana Jones

Image Copyright Walt Disney Company
The history in git is likely to be incomplete.
Yet, what can we do with it?
Problem #1: History is broken into repos
Linux History

- Many CVS repositories used before git
- Some code today not under git (eg. quilt)
- Lots of historical information spread in many mailing lists
Problem #2: git tracks changes per line
Warning

- The **author of a commit** (git parlance) is not necessarily the author of the code
  - Code imported from another source
  - Refactorings
    - Moving code
  - Code derived from other code
void alloc_report(void)
{
    REPORT(blob, struct blob);
    REPORT(tree, struct tree);
    REPORT(commit, struct commit);
    REPORT(tag, struct tag);
    REPORT(object, union any_object);
}
<table>
<thead>
<tr>
<th>Commit</th>
<th>Author</th>
<th>Date</th>
<th>Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>855419f</td>
<td>Linus Torvalds</td>
<td>19 Jun 2006</td>
<td>Add specialized object allocator</td>
</tr>
<tr>
<td>c335d74</td>
<td>nef</td>
<td>10 Jun 2014</td>
<td>alloc: include any-object allocator</td>
</tr>
<tr>
<td>225ea22</td>
<td>Ramsay Jones</td>
<td>13 Jul 2014</td>
<td>alloc.c: remove the alloc_raw_c call</td>
</tr>
<tr>
<td>c335d74</td>
<td>nef</td>
<td>10 Jun 2014</td>
<td>alloc: include any-object allocator</td>
</tr>
</tbody>
</table>

```c
void alloc_report(void)
{
    REPORT(blob, struct blob);
    REPORT(tree, struct tree);
    REPORT(commit, struct commit);
    REPORT(tag, struct tag);
    REPORT(object, union any_object);
}
```
void alloc_report(void)
{
  REPORT(blob, struct blob);
  REPORT(tree, struct tree);
  REPORT(commit, struct commit);
  REPORT(tag, struct tag);
  REPORT(object, union any_object);
}

Problem: git tracks changes per line but developers write code (tokens)
void alloc_report(void) {
  REPORT(blob, struct blob);
  REPORT(tree, struct tree);
  REPORT(commit, struct commit);
  REPORT(tag, struct tag);
  REPORT(object, union any_object);
}
void alloc_report(void)
{
    REPORT(blob, struct blob);
    REPORT(tree, struct tree);
    REPORT(commit, struct commit);
    REPORT(tag, struct tag);
    REPORT(object, union any_object);
}
void alloc_report(void)
{
    REPORT(blob, struct blob);
    REPORT(tree, struct tree);
    REPORT(commit, struct commit);
    REPORT(tag, struct tag);
    REPORT(object, union any_object);
}
All there sections have wrong attribution.
Evolutionary Views of VC Repos
file \rightarrow \text{filter(file)} \rightarrow \text{view(file)}

\text{to tokenize.}
void alloc_report(void)
{
    REPORT(blob, struct blob);
    REPORT(tree, struct tree);
    REPORT(commit, struct commit);
    REPORT(tag, struct tag);
    REPORT(object, union any_object);
}
Original Repo

c1

\[ \text{c2} \rightarrow \text{c4} \rightarrow \text{c5} \rightarrow \text{c3} \]

View Repo

\[ \text{view(c1)} \rightarrow \text{view(c2)} \rightarrow \text{view(c4)} \rightarrow \text{view(c3)} \rightarrow \text{view(c5)} \]
void alloc_report(void)
{
    REPORT(blob, struct blob);
    REPORT(tree, struct tree);
    REPORT(commit, struct commit);
    REPORT(tag, struct tag);
    REPORT(object, union any_object);
}

name|commit
argument_list|
expr_list
name|commit
argument_list|
expr_list
name|REPORT
argument_list|
expr_list
name|REPORT
argument_list|
expr_list
name|REPORT
argument_list|
expr_list
name|REPORT
argument_list|
expr_list
name|REPORT
argument_list|
expr_list
name|REPORT
argument_list|
expr_list
name|REPORT
argument_list|
expr_list
name|REPORT
argument_list|
expr_list
name|REPORT
argument_list|
expr_list
name|REPORT
argument_list|
expr_list
name|REPORT
argument_list|
expr_list
name|REPORT
argument_list|
expr_list
name|REPORT
argument_list|
expr_list
name|REPORT
argument_list|
expr_list
name|REPORT
argument_list|
expr_list
name|REPORT
argument_list|
expr_list
name|REPORT
argument_list|
expr_list
name|REPORT
argument_list|
expr_list
name|REPORT
argument_list|
expr_list
name|REPORT
argument_list|
expr_list
name|REPORT
argument_list|
expr_list
name|REPORT
argument_list|
expr_list
name|REPORT
argument_list|
expr_list
name|REPORT
argument_list|
expr_list
name|REPORT
argument_list|
expr_list
name|REPORT
argument_list|
expr_list
name|REPORT
argument_list|
expr_list
name|REPORT
argument_list|
expr_list
name|REPORT
argument_list|
expr_list
name|REPORT
argument_list|
expr_list
name|REPORT
argument_list|
expr_list
name|REPORT
argument_list|
expr_list
name|REPORT
argument_list|
expr_list
name|REPORT
argument_list|
expr_list
commit: push commit_index update into alloc_commit_node

Whenever we create a commit object via lookup_commit, we give it a unique index to be used with the commit-slab API. The theory is that any "struct commit" we create would follow this code path, so any such struct would get an index. However, callers could use alloc_commit_node() directly (and get multiple commits with index 0).

Let's push the indexing into alloc_commit_node so that it's hard for callers to get it wrong.

Signed-off-by: Jeff King <peff@peff.net>
Signed-off-by: Junio C Hamano <gitster@pobox.com>

1 parent c335d74  commit 969eba6341a5af08ac52c67e2646e2548ed05e23e3

Showing 2 changed files with 10 additions and 4 deletions.

--- alloc.c
@@ -47,10 +47,18 @@ union any_object {
 DEFINE_ALLOCATOR(blob, struct blob)
 DEFINE_ALLOCATOR(tree, struct tree)
-DEFINE_ALLOCATOR(commit, struct commit)
+DEFINE_ALLOCATOR(raw_commit, struct commit)
 DEFINE_ALLOCATOR(tag, struct tag)
 DEFINE_ALLOCATOR(object, union any_object)
Whenever we create a commit object via lookup_commit, we give it a unique index to be used with the commit-slab API. The theory is that any "struct commit" we create would follow this code path, so any such struct would get an index. However, callers could use alloc_commit_node() directly (and get multiple commits with index 0).

Let's push the indexing into alloc_commit_node so that it's hard for callers to get it wrong.

Signed-off-by: Jeff King <peff@peff.net>
Signed-off-by: Junio C Hamano <gitster@pobox.com>

commit: push commit_index update into alloc_commit_node

credit: 969eba6341a5af6ac52c67e26462548ed65e23e3  http://github.com/git/git/commit/969eba6341a5af6ac52c67e26462548ed65e23e3

peff committed with dmgerman on 10 Jun 2014

Showing 2 changed files with 35 additions and 16 deletions.

37 alloc.c

37 begin_macro
38 name|DEFINE_ALLOCATOR
39 argument_list|
40 -argument|commit
41 argument_list|
42 argument|struct commit
43
44 begin_macro
45 name|DEFINE_ALLOCATOR
46 argument_list|
47 +argument|raw_commit
48 argument_list|
49 argument|struct commit
Up to 4.7

<table>
<thead>
<tr>
<th>Language</th>
<th>files</th>
<th>blank</th>
<th>comment</th>
<th>code</th>
<th>#Comments</th>
<th>code</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>23,040</td>
<td>2,223,857</td>
<td>2,046,386</td>
<td>11,331,924</td>
<td>879,186</td>
<td>69,563,180</td>
</tr>
<tr>
<td>C/C++ Header</td>
<td>17,506</td>
<td>426,656</td>
<td>729,310</td>
<td>2,797,340</td>
<td>461,940</td>
<td>12,379,194</td>
</tr>
<tr>
<td></td>
<td>40,546</td>
<td>2,650,513</td>
<td>2,775,696</td>
<td>14,129,264</td>
<td>1,341,126</td>
<td>81,942,374</td>
</tr>
</tbody>
</table>

Persons in blame:

Line: 12,005
Token: 12,087
```c
static noinline_for_stack
int skip_atoi(const char **s)
{
    int i = 0;

    do {
        i = i*10 + (*((s)++) - '0');
    } while (isdigit(**s));

    return i;
}
```
Switch to demo
Aggregated information
Comparing token vs Line

- Does it make a difference to count contribution
  - Per line, vs
  - Per Token

- Eg:
  - Linux Kernel Report
Year of origin of code in Linux as of v4.7

Accumulated proportion

Proportion per year

1990 2000 2010

50% Y.T.
Year of origin of code in kernel as of v4.7
<table>
<thead>
<tr>
<th>index</th>
<th>token</th>
<th>ctokens</th>
<th>prop</th>
<th>accum</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>pre-git</td>
<td>4286384</td>
<td>5.15</td>
<td>5.15</td>
</tr>
<tr>
<td>2</td>
<td>alex deucher</td>
<td>2532472</td>
<td>3.04</td>
<td>8.19</td>
</tr>
<tr>
<td>3</td>
<td>larry finger</td>
<td>1424969</td>
<td>1.71</td>
<td>9.9</td>
</tr>
<tr>
<td>4</td>
<td>linus torvalds</td>
<td>1077539</td>
<td>1.29</td>
<td>11.19</td>
</tr>
<tr>
<td>5</td>
<td>jaroslav kysela</td>
<td>983822</td>
<td>1.18</td>
<td>12.37</td>
</tr>
<tr>
<td>6</td>
<td>peng tao</td>
<td>731861</td>
<td>0.88</td>
<td>13.25</td>
</tr>
<tr>
<td>7</td>
<td>andrew morton</td>
<td>728759</td>
<td>0.88</td>
<td>14.13</td>
</tr>
<tr>
<td>8</td>
<td>mark brown</td>
<td>648497</td>
<td>0.78</td>
<td>14.91</td>
</tr>
<tr>
<td>9</td>
<td>david howells</td>
<td>624218</td>
<td>0.75</td>
<td>15.66</td>
</tr>
<tr>
<td>10</td>
<td>hans verkuil</td>
<td>601867</td>
<td>0.72</td>
<td>16.38</td>
</tr>
<tr>
<td>11</td>
<td>ben skeggs</td>
<td>590972</td>
<td>0.71</td>
<td>17.09</td>
</tr>
<tr>
<td>12</td>
<td>david s. miller</td>
<td>541773</td>
<td>0.65</td>
<td>17.74</td>
</tr>
<tr>
<td>13</td>
<td>arend van spriel</td>
<td>501988</td>
<td>0.6</td>
<td>18.34</td>
</tr>
<tr>
<td>14</td>
<td>david daney</td>
<td>491898</td>
<td>0.59</td>
<td>18.93</td>
</tr>
<tr>
<td>15</td>
<td>alan cox</td>
<td>469412</td>
<td>0.56</td>
<td>19.57</td>
</tr>
<tr>
<td>16</td>
<td>laurent pinchart</td>
<td>461247</td>
<td>0.55</td>
<td>20.05</td>
</tr>
<tr>
<td>17</td>
<td>christoph hellwig</td>
<td>427959</td>
<td>0.51</td>
<td>20.56</td>
</tr>
<tr>
<td>18</td>
<td>mauro carvalho chehab</td>
<td>404112</td>
<td>0.49</td>
<td>21.05</td>
</tr>
<tr>
<td>19</td>
<td>johannes berg</td>
<td>401906</td>
<td>0.48</td>
<td>21.53</td>
</tr>
<tr>
<td>20</td>
<td>takashi iwai</td>
<td>367443</td>
<td>0.44</td>
<td>21.97</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>index</th>
<th>token</th>
<th>clines</th>
<th>prop</th>
<th>accum</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>pre-git</td>
<td>745226</td>
<td>3.81</td>
<td>3.81</td>
</tr>
<tr>
<td>2</td>
<td>alex deucher</td>
<td>609927</td>
<td>3.12</td>
<td>6.93</td>
</tr>
<tr>
<td>3</td>
<td>larry finger</td>
<td>295550</td>
<td>1.51</td>
<td>8.44</td>
</tr>
<tr>
<td>4</td>
<td>linus torvalds</td>
<td>247626</td>
<td>1.27</td>
<td>9.7</td>
</tr>
<tr>
<td>5</td>
<td>jaroslav kysela</td>
<td>185692</td>
<td>0.95</td>
<td>10.65</td>
</tr>
<tr>
<td>6</td>
<td>david howells</td>
<td>181493</td>
<td>0.93</td>
<td>11.58</td>
</tr>
<tr>
<td>7</td>
<td>mark brown</td>
<td>156540</td>
<td>0.8</td>
<td>12.38</td>
</tr>
<tr>
<td>8</td>
<td>peng tao</td>
<td>149764</td>
<td>0.77</td>
<td>13.14</td>
</tr>
<tr>
<td>9</td>
<td>andrew morton</td>
<td>135377</td>
<td>0.69</td>
<td>13.83</td>
</tr>
<tr>
<td>10</td>
<td>joe perches</td>
<td>125900</td>
<td>0.64</td>
<td>14.48</td>
</tr>
<tr>
<td>11</td>
<td>ben skeggs</td>
<td>123504</td>
<td>0.63</td>
<td>15.11</td>
</tr>
<tr>
<td>12</td>
<td>david daney</td>
<td>123350</td>
<td>0.63</td>
<td>15.74</td>
</tr>
<tr>
<td>13</td>
<td>hans verkuil</td>
<td>123131</td>
<td>0.63</td>
<td>16.37</td>
</tr>
<tr>
<td>14</td>
<td>mauro carvalho chehab</td>
<td>119616</td>
<td>0.61</td>
<td>16.98</td>
</tr>
<tr>
<td>15</td>
<td>david s. miller</td>
<td>118700</td>
<td>0.61</td>
<td>17.59</td>
</tr>
<tr>
<td>16</td>
<td>alan cox</td>
<td>106285</td>
<td>0.54</td>
<td>18.13</td>
</tr>
<tr>
<td>17</td>
<td>johannes berg</td>
<td>105760</td>
<td>0.54</td>
<td>18.67</td>
</tr>
<tr>
<td>18</td>
<td>christoph hellwig</td>
<td>103596</td>
<td>0.53</td>
<td>19.2</td>
</tr>
<tr>
<td>19</td>
<td>laurent pinchart</td>
<td>99141</td>
<td>0.51</td>
<td>19.71</td>
</tr>
<tr>
<td>20</td>
<td>takashi iwai</td>
<td>96905</td>
<td>0.5</td>
<td>20.2</td>
</tr>
<tr>
<td>index</td>
<td>personid</td>
<td>token</td>
<td>clines</td>
<td>prop</td>
</tr>
<tr>
<td>-------</td>
<td>----------------</td>
<td>---------------</td>
<td>--------</td>
<td>------</td>
</tr>
<tr>
<td>1</td>
<td>steven rostedt</td>
<td>27274</td>
<td>10.38</td>
<td>10.38</td>
</tr>
<tr>
<td>2</td>
<td>peter zijlstra</td>
<td>19704</td>
<td>7.5</td>
<td>17.88</td>
</tr>
<tr>
<td>3</td>
<td>thomas gleixner</td>
<td>14139</td>
<td>5.38</td>
<td>23.26</td>
</tr>
<tr>
<td>4</td>
<td>paul e. mckenney</td>
<td>13353</td>
<td>5.08</td>
<td>28.34</td>
</tr>
<tr>
<td>5</td>
<td>tejun heo</td>
<td>11482</td>
<td>4.37</td>
<td>32.71</td>
</tr>
<tr>
<td>6</td>
<td>ingo molnar</td>
<td>9522</td>
<td>3.62</td>
<td>36.33</td>
</tr>
<tr>
<td>7</td>
<td>jason wessel</td>
<td>8216</td>
<td>3.13</td>
<td>39.46</td>
</tr>
<tr>
<td>8</td>
<td>frederic weisbecker</td>
<td>6961</td>
<td>2.65</td>
<td>42.1</td>
</tr>
<tr>
<td>9</td>
<td>rafael j. wysocki</td>
<td>6264</td>
<td>2.38</td>
<td>44.49</td>
</tr>
<tr>
<td>10</td>
<td>alexei starovoitov</td>
<td>6231</td>
<td>2.37</td>
<td>46.86</td>
</tr>
<tr>
<td>11</td>
<td>tom zanussi</td>
<td>6144</td>
<td>2.34</td>
<td>49.2</td>
</tr>
<tr>
<td>12</td>
<td>oleg nesterov</td>
<td>5314</td>
<td>2.02</td>
<td>51.22</td>
</tr>
<tr>
<td>13</td>
<td>eric w. biederman</td>
<td>3987</td>
<td>1.52</td>
<td>52.74</td>
</tr>
<tr>
<td>14</td>
<td>rusty russell</td>
<td>3558</td>
<td>1.35</td>
<td>54.09</td>
</tr>
<tr>
<td>15</td>
<td>masami hiramatsu</td>
<td>3166</td>
<td>1.2</td>
<td>55.29</td>
</tr>
<tr>
<td>16</td>
<td>al viro</td>
<td>3000</td>
<td>1.14</td>
<td>56.44</td>
</tr>
<tr>
<td>17</td>
<td>pre-git</td>
<td>2737</td>
<td>1.04</td>
<td>57.48</td>
</tr>
<tr>
<td>18</td>
<td>john stultz</td>
<td>2576</td>
<td>0.98</td>
<td>58.46</td>
</tr>
<tr>
<td>19</td>
<td>dave young</td>
<td>2546</td>
<td>0.97</td>
<td>59.43</td>
</tr>
<tr>
<td>20</td>
<td>jiri olsa</td>
<td>2489</td>
<td>0.95</td>
<td>60.37</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>index</th>
<th>personid</th>
<th>token</th>
<th>ctokens</th>
<th>prop</th>
<th>accum</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>steven rostedt</td>
<td>94860</td>
<td>9.81</td>
<td>9.81</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>peter zijlstra</td>
<td>80844</td>
<td>8.36</td>
<td>18.18</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>paul e. mckenney</td>
<td>47737</td>
<td>4.94</td>
<td>23.12</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>thomas gleixner</td>
<td>43543</td>
<td>4.51</td>
<td>27.62</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>tejun heo</td>
<td>33074</td>
<td>3.42</td>
<td>31.05</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>jason wessel</td>
<td>32133</td>
<td>3.32</td>
<td>34.37</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>ingo molnar</td>
<td>29380</td>
<td>3.04</td>
<td>37.41</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>frederic weisbecker</td>
<td>28123</td>
<td>2.91</td>
<td>40.32</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>alexei starovoitov</td>
<td>27944</td>
<td>2.89</td>
<td>43.21</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>tom zanussi</td>
<td>25010</td>
<td>2.59</td>
<td>45.8</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>rafael j. wysocki</td>
<td>21292</td>
<td>2.2</td>
<td>48.0</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>oleg nesterov</td>
<td>18696</td>
<td>1.93</td>
<td>49.94</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>eric w. biederman</td>
<td>17823</td>
<td>1.84</td>
<td>51.78</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>rusty russell</td>
<td>14845</td>
<td>1.54</td>
<td>53.32</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>al viro</td>
<td>14210</td>
<td>1.47</td>
<td>54.79</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>masami hiramatsu</td>
<td>13423</td>
<td>1.39</td>
<td>56.18</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>pre-git</td>
<td>11244</td>
<td>1.16</td>
<td>57.34</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>srikar dronamraju</td>
<td>10757</td>
<td>1.11</td>
<td>58.45</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>eric paris</td>
<td>10311</td>
<td>1.07</td>
<td>59.52</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>dave young</td>
<td>10197</td>
<td>1.06</td>
<td>60.57</td>
<td></td>
</tr>
</tbody>
</table>
Many small changes

- 9.5% of non-merge commits added 3 or less tokens and removed 3 or less tokens
Warning

• cregit does not tell you where the code comes from
  – Nor who authored it
• It only gives you one more layer of information

• True provenance might never be feasible (in practice)
cregit: what does it tell us

- On the large
  - Token and Line are equivalent

- On the small
  - Provide a fine grained view of the evolution of the code
What is next? Who reviewed the code

- We are currently adding the review information in mailings lists to the kernel
  - We have excellent heuristics to match email reviews to a given commit
What is next: cregit future

- Enhance the commit/change information
  - **Origin analysis:** where did the code come from?
  - **Who:** truly created/authored the code?
  - Is the new code a **derivation** of other code?
Code is a sequence of changes

- Each change (the commit)
  - Has one or many copyright-authors
  - It might be based on previously written code
    - Moved code
    - Copied code
    - Improved code
  - Its history might be recorded somewhere else
Code is a sequence of changes

- *The current* version is the composition of commits from beginning of (project) time

- A commit is a “*function*” that *changes* the *previous code* into the *new code*

- This *function* might have *multiple inputs*
  - Previous version of the code
  - External code
    - This code might be created in a different repository and *might already have a rich history*
  - Knowledge from the developer (the creative part)
  - Generated code
What is next: cregit future

- Create new **views** of history
  - As long as it is diffable, cregit can track its evolution

- Use cregit as a **tool** to inspect **history**
  - Example: (switch to github)
cregit: identifying contributors of source code

Daniel M German
University of Victoria
dmg@uvic.ca

Kate Stewart
Linux Foundation

Bram Adams
Polytechnique of Montreal
cregit: identifying contributors of source code

Daniel M German
University of Victoria
dmg@uvic.ca

Kate Stewart
Linux Foundation

Bram Adams
Polytechnique of Montreal
cregit: identifying contributors of source code

Daniel M German
University of Victoria
dmg@uvic.ca

Kate Stewart
Linux Foundation

Bram Adams
Polytechnique of Montreal

```c
sprint_backtrace(sym, value);
else if (*fmt != 'f' && *fmt != 's')
sprint_symbol(sym, value);
else
    sprint_symbol_no_offset(sym, value);

return string(buf, end, sym, spec);
#else
    return special_hex_number(buf, end, value, sizeof(void *));
#endif
```

Contributors

<table>
<thead>
<tr>
<th>Person</th>
<th>Tokens</th>
<th>Prop</th>
</tr>
</thead>
<tbody>
<tr>
<td>linus torvalds</td>
<td>65</td>
<td>44.83%</td>
</tr>
<tr>
<td>joe perches</td>
<td>39</td>
<td>26.90%</td>
</tr>
<tr>
<td>frederic weisbecker</td>
<td>16</td>
<td>11.83%</td>
</tr>
<tr>
<td>namhyung kim</td>
<td>13</td>
<td>8.97%</td>
</tr>
</tbody>
</table>
cregit: identifying contributors of source code

Daniel M German
University of Victoria
dmg@uvic.ca

Kate Stewart
Linux Foundation

Bram Adams
Polytechnique of Montreal

Code is a sequence of changes

- **The current** version is the composition of commits from beginning of (project) time

- A commit is a “function” that changes the previous code into the new code

- This function might have multiple inputs
  - Previous version of the code
  - External code
    - This code might be created in a different repository and might already have a rich history
  - Knowledge from the developer (the creative part)
  - Generated code

<table>
<thead>
<tr>
<th>Contributors</th>
<th>Person</th>
<th>Tokens</th>
<th>Prop</th>
</tr>
</thead>
<tbody>
<tr>
<td>linus torvalds</td>
<td>linus torvalds</td>
<td>65</td>
<td>44.83</td>
</tr>
<tr>
<td>joe perches</td>
<td>joe perches</td>
<td>39</td>
<td>26.90</td>
</tr>
<tr>
<td>frederic weisbecker</td>
<td>frederic weisbecker</td>
<td>16</td>
<td>11.83</td>
</tr>
<tr>
<td>namhyung kim</td>
<td>namhyung kim</td>
<td>13</td>
<td>8.97</td>
</tr>
</tbody>
</table>