Praktikum angewandte Systemsoftwaretechnik

Alexander Würstlein

Lehrstuhl Informatik 4

2016-06-30

git bisect

- Werkzeug zur Fehlersuche in einem git-Repository
- „Welche Revision hat den Fehler eingeführt?“
- Voraussetzungen:
  - 1 Revision in der der Fehler auftritt (üblicherweise die aktuelle)
  - 1 Revision in der der Fehler (noch nicht) auftritt
  - manueller oder automatischer Test für den Fehler
  - möglichst viele Revisionen sollten testbar sein (z.B. compilieren)

Bisektion

- mathematisch: Nullstellensuche in Intervall auf stetiger Funktion f
- f stetig auf $]a; c[\wedge f(a) < 0 \wedge f(c) > 0 \Rightarrow \exists b, a < b < c : f(b) = 0$
- hier: „Nullstelle“ ist Übergang zwischen „geht“ und „geht nicht“
- `git bisect` findet garantiert *einen* Übergang
- aber: es könnte Mehrere geben

Verfahren (beinahe binäre Suche)

- wähle Intervall $]a; c[$, so dass Fehler in c, kein Fehler in a
- wähle b „in der Mitte“ zwischen a und c
- prüfe b auf Fehler
  - Fehler: von vorne mit Intervall a, b
  - kein Fehler: von vorne mit Intervall b, c
- fertig, wenn Intervall leer

Bedienung

- Anfang: `git bisect start; git bisect bad; git bisect good 04711deadbeef`
- startet bisect mit HEAD als „kaputt“ markiert und „04711deadbeef“ als „funktionierend“
- git checkt automatisch Version dazwischen aus, testen und...
  - `git bisect good`
  - `git bisect bad`
  - `git bisect skip`
- ... wiederholen bis fertig
- Übersicht: `git bisect log; git bisect visualize`

spart Zeit...

- n Revisionen im Intervall
- jeder Schritt halbiert Intervall (ausser bei „skip“)
- Schritte im besten Fall: $\lceil \log_2 n \rceil$
- 10 Schritte für etwa 1000 Revisionen, 20 Schritte für 1 Mio.
- Schritte im schlechtesten Fall: $n - 1$
- schlechtester Fall, wenn keine Revision baut/testbar ist
- Moral: nur compilierenden Code ins „richtige“ Repository

spart noch mehr Zeit...

- `git bisect start -- src/subsystem/subsubsystem`: Nur Teilbäume betrachten
- nonplusultra: automagisches git bisect
- benötigt
  - automatischen Test
  - funktionierendes Build-Skript
  - kleines Skript das beides aufruft
- `git bisect run ./test.sh`
- Moral: automatische Testsuite und funktionierende schnelle Build-Skripte sind toll
- `git clone /proj/i4passt/git/bisect-demo.git ; git checkout i18n ; git checkout master`