Go 语言发布历史回顾

杨文

https://golang.design/history

Go 1.0 2012.3.28

兼容性说明文档说明，Go 的未来版本会确保向后兼容性，不会破坏现有程序。

此版本已经包含 go tool pprof 命令，同时还包含了 go vet 命令，它可以报告程序包中可能存在的错误。

pprof 是 Google pprof C++ 分析器的一个变种，

Go 1.1 2013.5.13

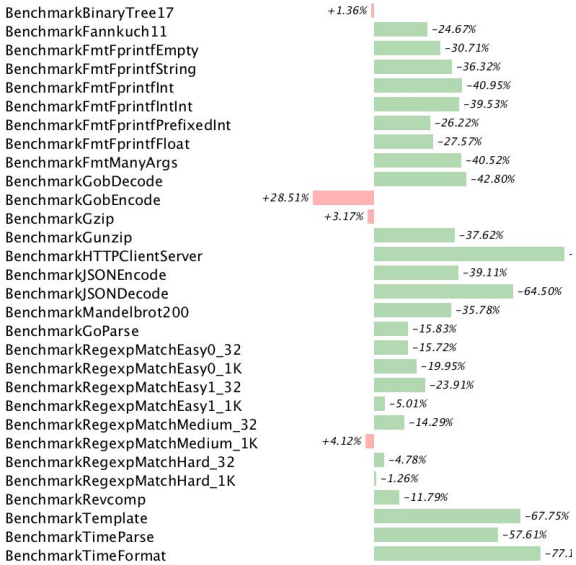
该版本 Go 主要是增强语言特性（包括编译器、垃圾回收机制、map、goroutine 调度器）与性能。

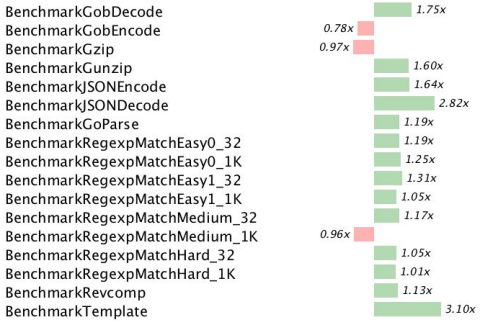
该版本内置了竞态检测器，Go 语言必不可少的工具。

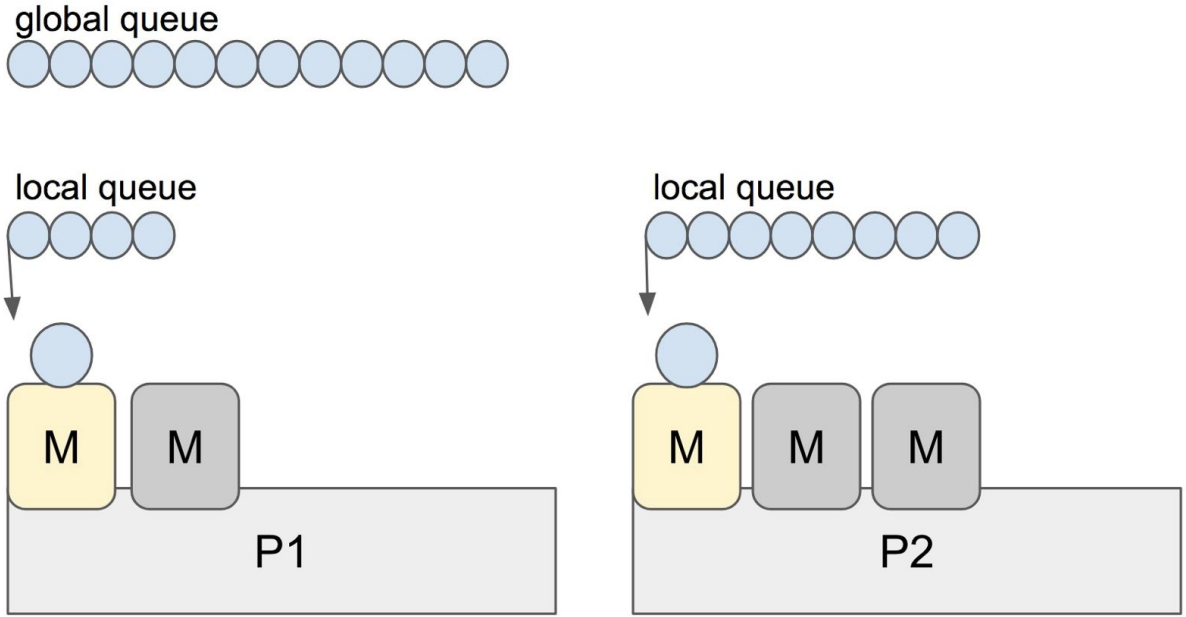
重新编写后的 Go 的调度器性能有了显著提高。

Baseline Benchmarks

linux/amd64 Lenovo x220, 8Gb ram, Core i5 2.5Ghz
Ubuntu 13.04







Go 1.2 2013.12.01

test 命令支持代码覆盖率报告，并提供新的 go tool cover 命令输出代码测试覆盖率的统计信息。还提供了 UI 界面显示代码覆盖率的详细信息。（精确到代码行）

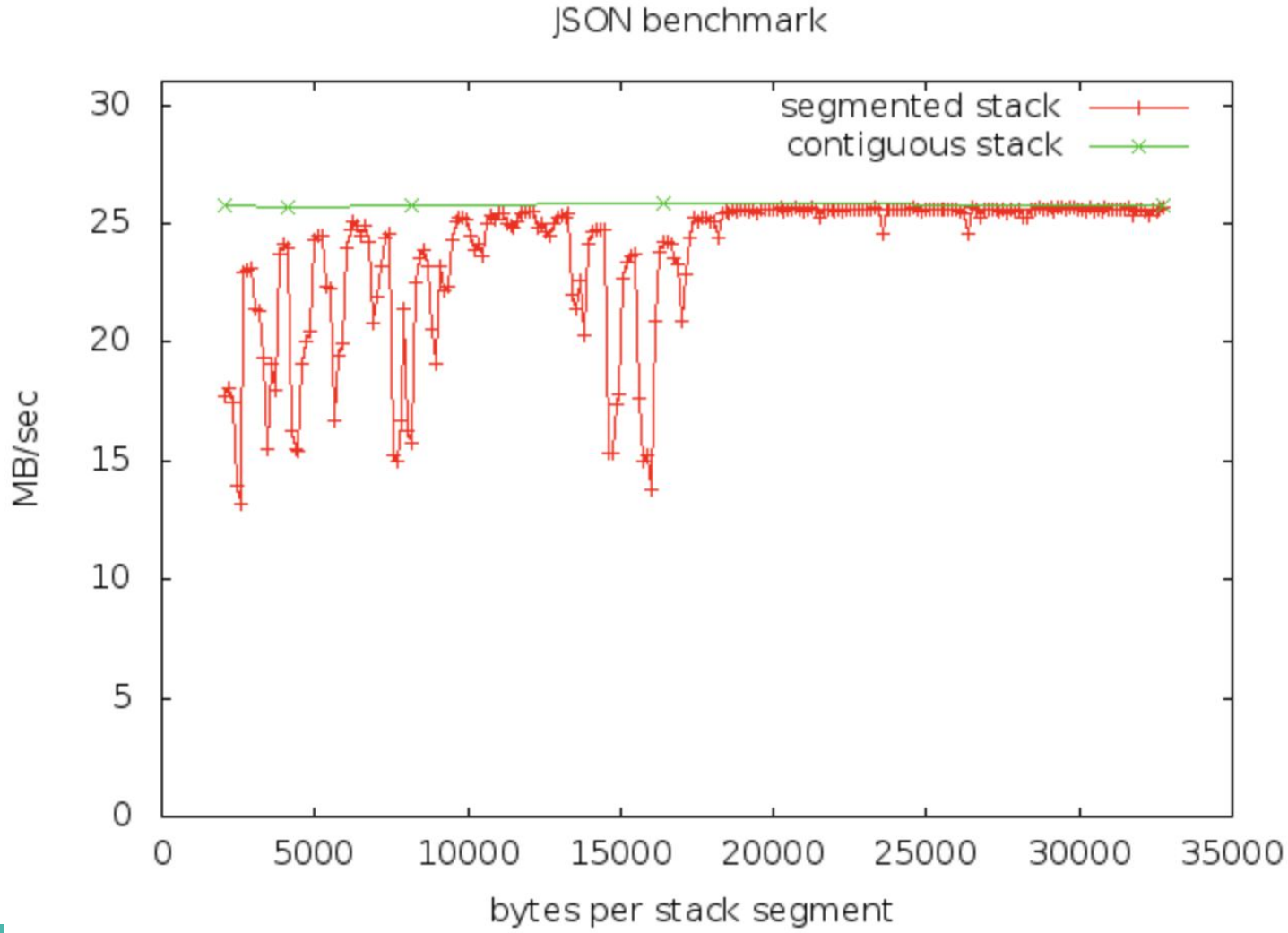
Go 1.3 2014.06.18

该版本重要改善了堆栈管理。堆栈现在会分配连续的内存片段，提供了分配效率。使得 Go 语言在下个版本中将堆栈大小减少 2KB

该版本改进了某些组件中堆栈的错误拆分所导致的性能下降问题，此类问题会在堆栈密集分配/释放状态下出现。（引入连续堆栈的机制修复了这类组件的性能问题）

该版本发布了 sync pool 组件。（通过 pool 可以复用代码结构，减少分配资源的数量，并且它作为后续 Go 生态系统中许多改进的最底层支持，比方说，标准库的 encoding/json 和 net/http，zap，gin 等等。

该版本还改进了 channel 的实现，提升了它的性能。


JSON benchmark
segmented stack
contiguous stack
MB/sec
bytes per stack segment
0
5
10
15
20
25
30
0
5000
10000
15000
20000
25000
30000
35000

Go 1.2 与 Go 1.3 的基准测试对比

```
name                 old time/op  new time/op  delta
SelectUncontended     216ns ± 1%   179ns ± 1%  -16.94%  (p=0.029 n=4+4)
SelectContended       211ns ± 2%   183ns ± 1%  -13.27%  (p=0.029 n=4+4)
SelectNonblock       92.8ns ± 0%  93.7ns ± 3%     ~     (p=1.000 n=4+4)
ChanSync              120ns ± 1%   115ns ± 1%   -4.17%  (p=0.029 n=4+4)
ChanProdCons0         119ns ± 2%   114ns ± 0%   -4.20%  (p=0.029 n=4+4)
ChanProdCons10       70.5ns ± 1%  71.8ns ± 1%   +1.84%  (p=0.029 n=4+4)
ChanProdCons100      56.0ns ± 0%  56.6ns ± 1%   +1.07%  (p=0.029 n=4+4)
ChanProdConsWork0     556ns ± 3%   430ns ± 1%  -22.68%  (p=0.029 n=4+4)
ChanProdConsWork10    486ns ± 2%   373ns ± 0%  -23.25%  (p=0.029 n=4+4)
ChanProdConsWork100   462ns ± 0%   352ns ± 0%  -23.89%  (p=0.029 n=4+4)
ChanCreation          432ns ± 0%   292ns ± 1%  -32.60%  (p=0.029 n=4+4)
ChanSem              52.2ns ± 0%  53.0ns ± 0%   +1.53%  (p=0.029 n=4+4)
```

Go 1.4 2014.12.10

该版本发布了 Android 官方支持包 golang.org/x/mobile，可以使用 Go 代码编写简单的 Android 应用。

该版本把之前很多 C 和汇编语言编写的运行时转换为 Go 实现。

该版本由于使用了更精确的垃圾收集器，堆栈大小减少了 10~30%。

Go 的项目代码管理工具从 Mercurial 切换为 Git，项目也从 Google Code 迁移到了 GitHub。

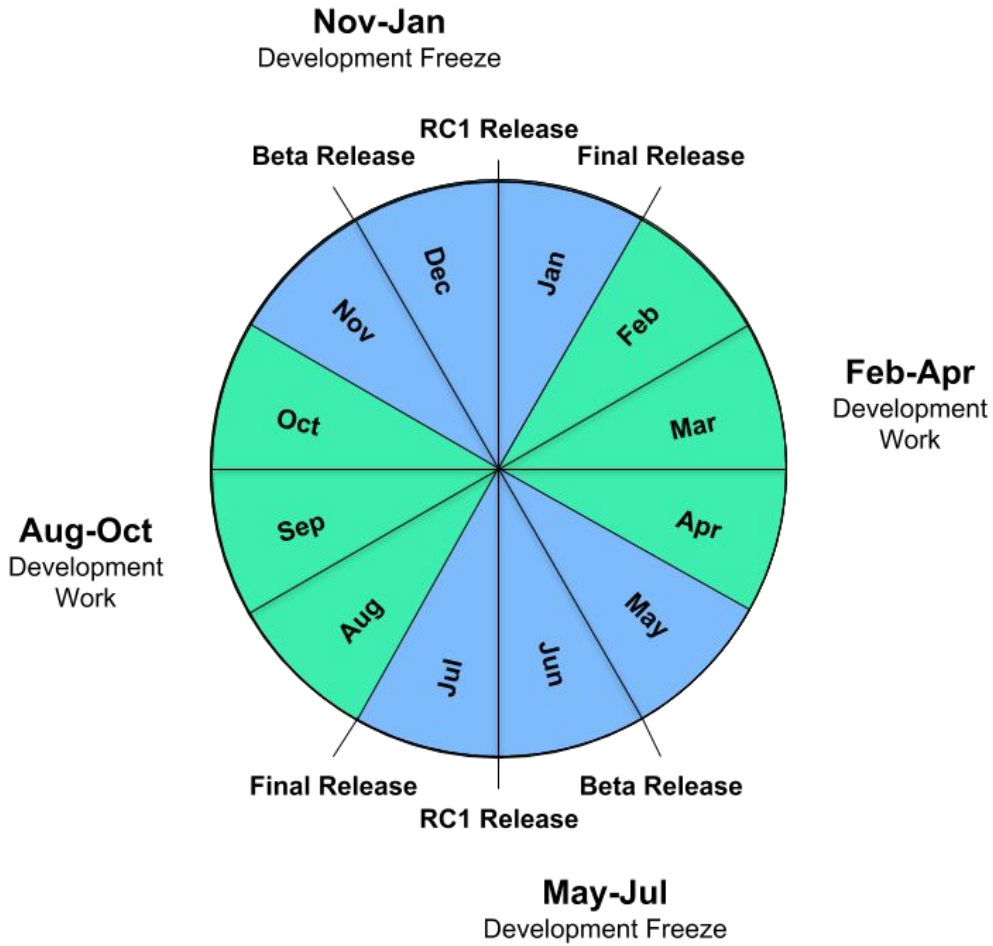
该版本还发布了 go generate 命令，扫描 //go:generate 指令提供的信息生成代码，简化了代码生成方式。

Go 1.5 2015.8.19

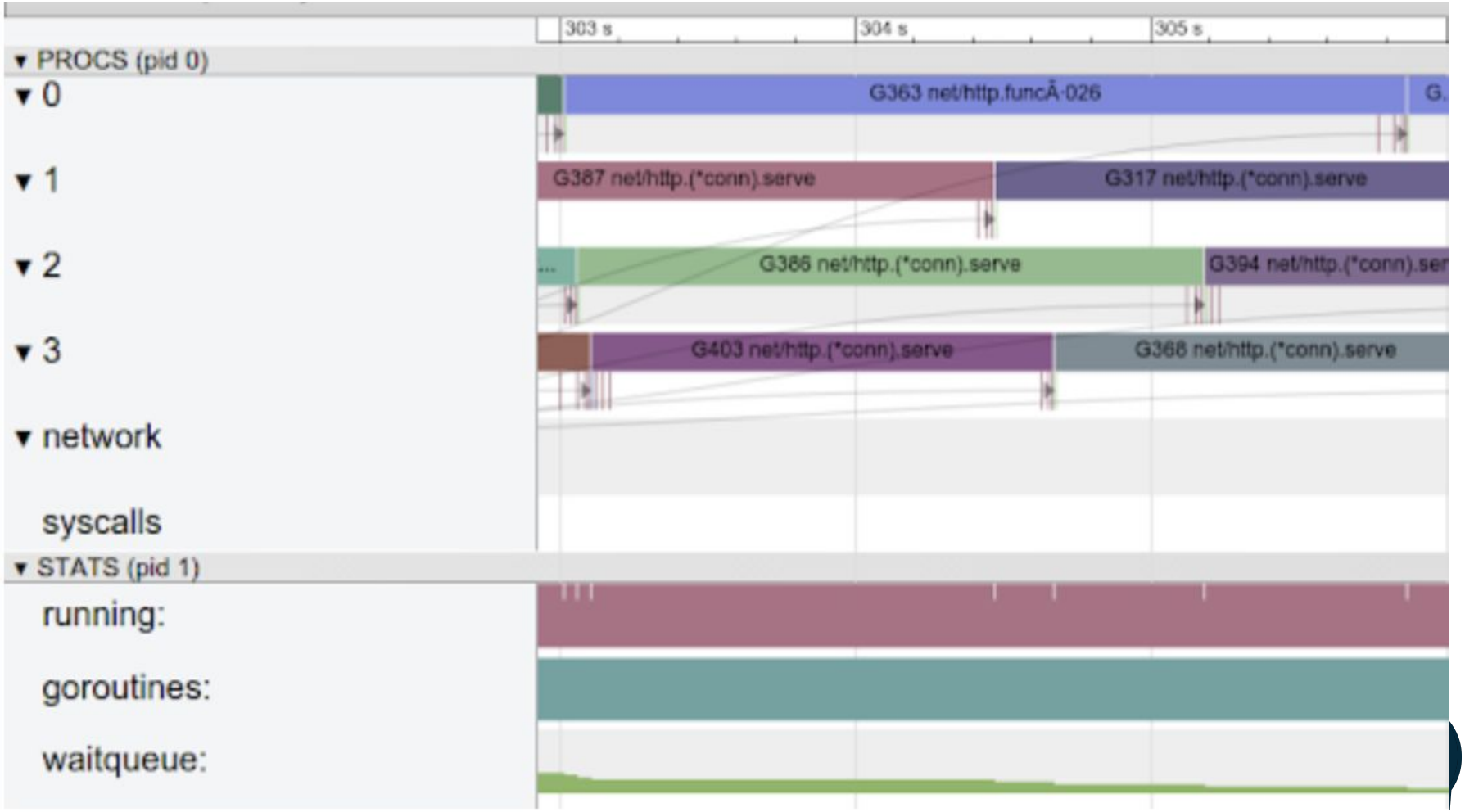
从该版本开始，Go 的发布时间为每年的8月和2月。

该版本完全重新设计和实现了垃圾回收器，基于并发的回收器，使得垃圾回收延迟被显著降低。

该版本还发布了执行追踪记录，可以通过 go tool trace 命令获取信息。追踪信息可在测试或运行期间生成，并展示在浏览器窗口里。


Nov-Jan
Development Freeze
RC1 Release
Beta Release
Final Release
Dec
Jan
Feb
Nov
Mar
Oct
Apr
Sep
May
Aug
Jun
Jul
Feb-Apr
Development
Work
Aug-Oct
Development
Work
Final Release
Beta Release
RC1 Release
May-Jul
Development Freeze

Latency (Milliseconds) 1.4 - 1.5 (Aug ‘15)

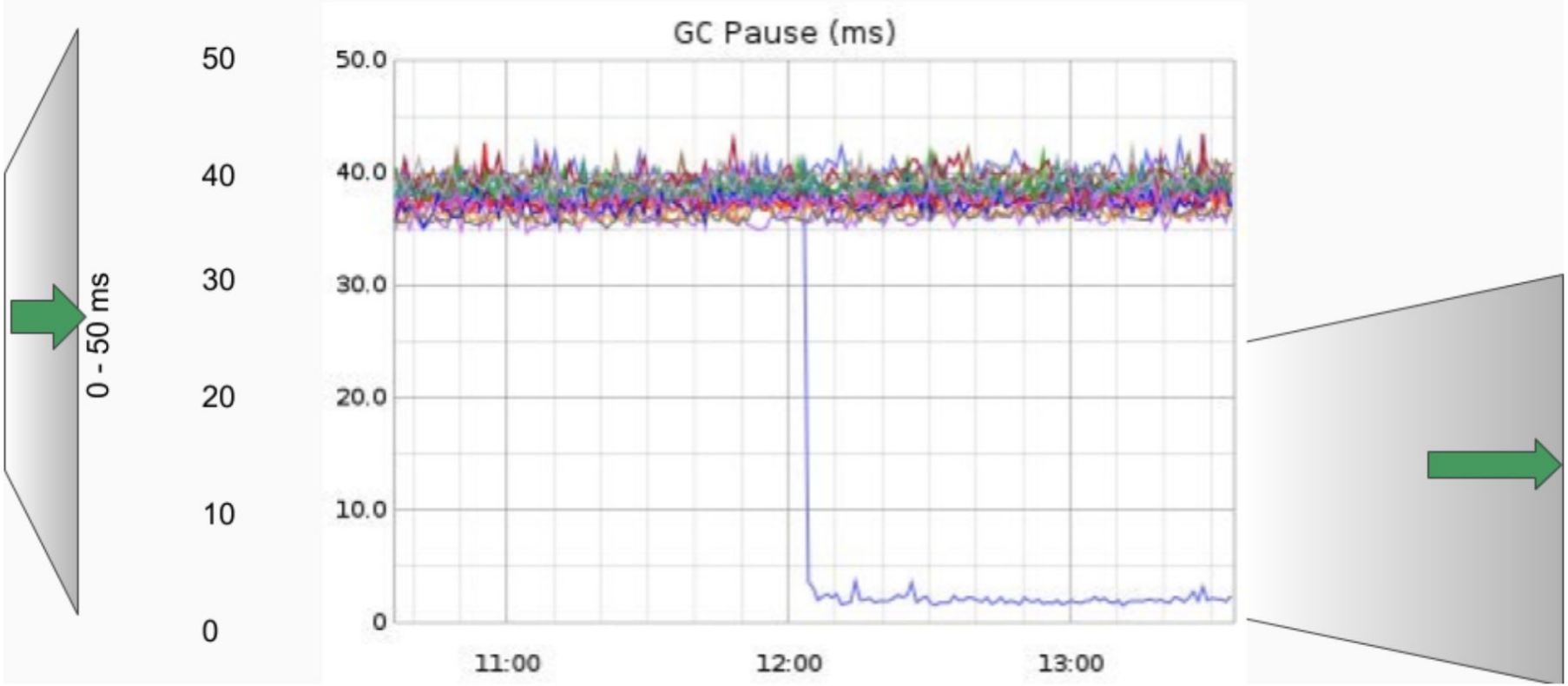

303 s
304 s
305 s
▾ PROCS (pid 0)
▾ 0
G363 net/http.funcÃ·026
▾ 1
G387 net/http.(*conn).serve
G317 net/http.(*conn).serve
▾ 2
G386 net/http.(*conn).serve
G394 net/http.(*conn).ser
▾ 3
G403 net/http.(*conn).serve
G368 net/http.(*conn).serve
▾ network
syscalls
▾ STATS (pid 1)
running:
goroutines:
waitqueue:

Go 1.6 2016.02.17

该版本主要是在 HTTPS 情况下，增加对 HTTP/2 协议的默认支持。

该版本再次降低了垃圾回收器的延迟。

Latency (Milliseconds) 1.5 - 1.6 (Mar ‘16)



Go 1.7 2016.08.15

该版本发布了 Context 包，可以为用户提供处理超时和任务取消的机制。

该版本优化了编译工具，加速了编译过程。

该版本还降低了编译后二进制文件的大小，幅度可达 20~30%。

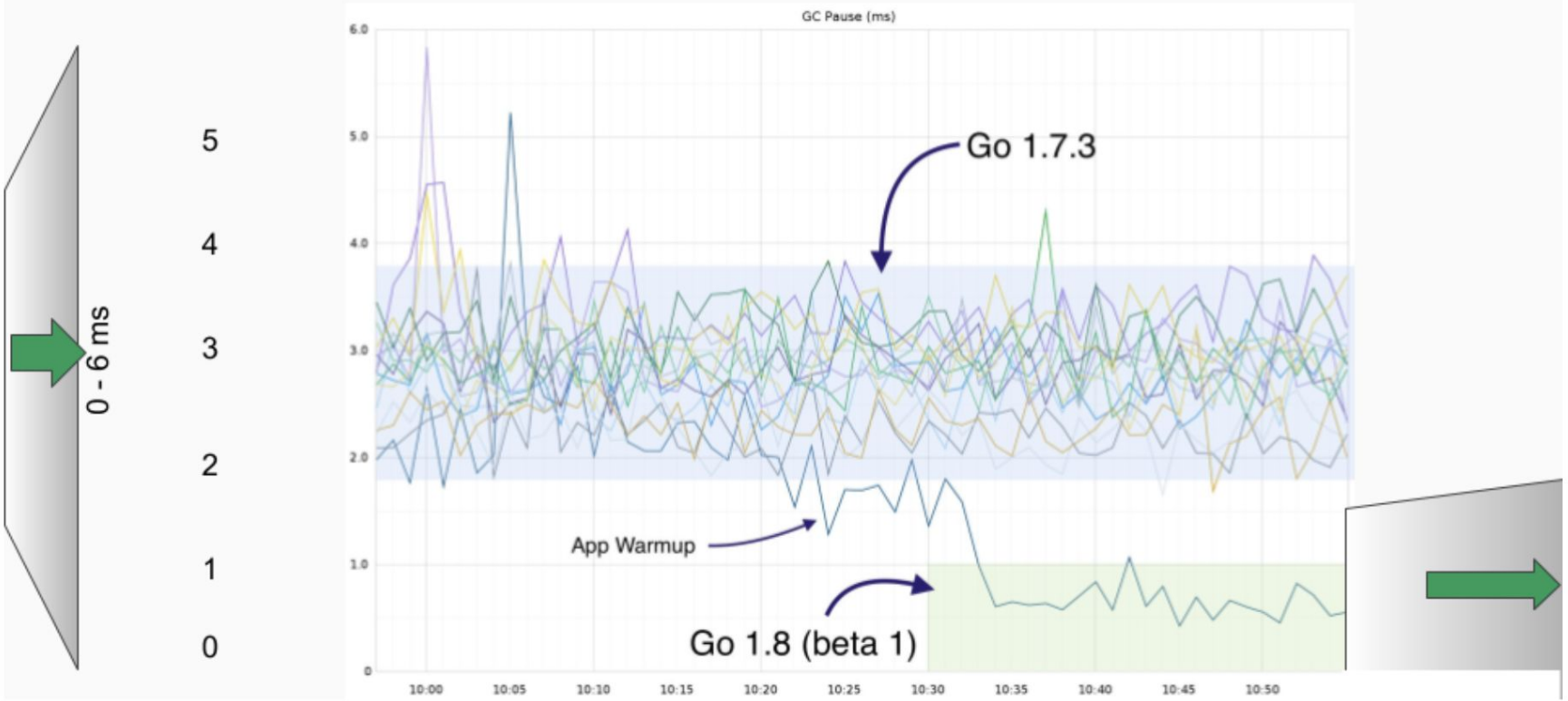
Go 1.8 2017.02.16

该版本改进了垃圾回收器，使得两次垃圾回收的暂停时间减小到了毫秒级。

该版本还修复了一个问题，使得在通常情况下暂停时间能控制在 100 微妙左右。

该版本还改进了 defer 函数。

Latency (Milliseconds) 1.7 - 1.8 (Mar ‘17)



```
name          old time/op  new time/op  delta
Defer-4       99.0ns ± 9%  52.4ns ± 5%  -47.04%  (p=0.000 n=9+10)
Defer10-4     90.6ns ±13%  45.0ns ± 3%  -50.37%  (p=0.000 n=10+10)
```

Go 1.9 2017.08.24

该版本增加了类型别名。

该版本 sync 包增加了保证并发访问安全性的 Map 类型。

Go 1.10 2018.02.16

该版本 test 包增加了新的智能缓存机制，自动跳过未做更改的代码的相关测试用例，节省开发人员运行测试套件的时间。

该版本还对 go build 命令缓存最近构建过的包，从而加快构建速度。

虽然不包含垃圾回收器的实质性改动，但是为它重新定义了 SLO（服务级别目标）

SLOs then and now

/I.

2014	2018
25% of the total CPU	25% of the CPU *during* GC cycle
Heap 2X live heap	Heap 2X live heap or max heap
10 ms STW pause every 50 ms	Two <500 µs STW pauses per GC
Goroutines allocation ∝ GC assists	Goroutines allocation ∝ GC assists
	Minimal GC assists in steady state

Go 1.11 2018.08.24

该版本引入 Go Modules。

该版本增加了实验性的 WebAssembly 支持，帮助开发人员将 Go 程序编译为兼容四个主要 web 浏览器的二进制程序。

Go 1.12 2019.02.25

该版本在 analysis 包的基础上重写了 go vet 命令，有了更大的灵活性，允许开发人员编写自己的代码检查工具。

Go 1.13 2019.09.03

该版本改进了 sync.Pool，使得池中的资源不会在垃圾回收的实话被清除。（通过新机制里引入的缓存，两次垃圾回收之间没有被使用过的实例才会被清除。）

该版本重写了逃逸分析逻辑，使得 Go 程序减少了堆上的分配次数。

```
name       old alloc/op      new alloc/op      delta
Template         39.0MB ± 0%       38.6MB ± 0%  -1.04%  (p=0.000 n=10+10)
Unicode          28.3MB ± 0%       28.3MB ± 0%  -0.08%  (p=0.000 n=10+10)
GoTypes           132MB ± 0%        131MB ± 0%  -0.77%  (p=0.000 n=10+9)
Compiler          625MB ± 0%        619MB ± 0%  -0.97%  (p=0.000 n=10+10)
SSA              2.04GB ± 0%       2.00GB ± 0%  -2.11%  (p=0.000 n=10+10)
Flate            24.2MB ± 0%       24.0MB ± 0%  -1.05%  (p=0.000 n=10+10)
GoParser         29.1MB ± 0%       28.8MB ± 0%  -1.19%  (p=0.000 n=10+10)
Reflect          84.6MB ± 0%       83.5MB ± 0%  -1.24%  (p=0.000 n=10+10)
Tar              36.9MB ± 0%       36.5MB ± 0%  -1.05%  (p=0.000 n=9+10)
XML              48.4MB ± 0%       47.7MB ± 0%  -1.43%  (p=0.000 n=10+10)

name       old allocs/op     new allocs/op     delta
Template          382k ± 0%         380k ± 0%  -0.60%  (p=0.000 n=9+10)
Unicode           341k ± 0%         341k ± 0%  -0.05%  (p=0.000 n=10+10)
GoTypes          1.36M ± 0%        1.36M ± 0%  -0.20%  (p=0.000 n=10+9)
Compiler         5.73M ± 0%        5.69M ± 0%  -0.60%  (p=0.000 n=10+10)
SSA              16.9M ± 0%        16.6M ± 0%  -1.49%  (p=0.000 n=10+9)
Flate             237k ± 0%         235k ± 0%  -0.95%  (p=0.000 n=10+10)
GoParser          302k ± 0%         302k ± 0%  -0.18%  (p=0.000 n=10+10)
Reflect           986k ± 0%         976k ± 0%  -0.95%  (p=0.000 n=10+10)
Tar               356k ± 0%         353k ± 0%  -0.80%  (p=0.000 n=8+10)
XML               441k ± 0%         437k ± 0%  -1.09%  (p=0.000 n=10+10)
```

Go 1.14 2020.02.25

该版本主要改进了 toolchain, runtime, libraries。

该版本支持 go Modules 在生产环境使用。

Go 1.15 2020.08.11

Go 链接器有了实质上的提升

提升了在高内核系统下的小对象内存分配

废弃了 X.509 CommonName

GOPROXY 限制已经支持在返回错误时跳过代理

增加了一个全新的嵌入式 tzdata 包

Go 1.16 2021.02.16