




# Golang Developer. Professional

[otus.ru](https://otus.ru)

 Проверить, идет ли запись

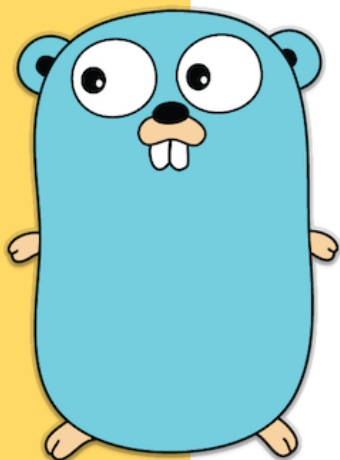
# Меня хорошо видно && слышно?

 Ставим “+”, если все хорошо  
“-”, если есть проблемы

Тема вебинара

# Go внутри. Планировщик

Алексей Романовский



# Правила вебинара



Активно  
участвуем



Off-topic обсуждаем  
в учебной группе

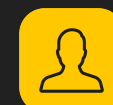


Задаем вопрос  
в чат или голосом



Вопросы вижу в чате,  
могу ответить не сразу

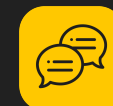
## Условные обозначения



Индивидуально



Время, необходимое  
на активность



Пишем в чат



Говорим голосом



Документ



Ответьте себе или  
задайте вопрос

# О чем будем говорить

- Планировщик
  - Что такое планировщик.
  - Особенности работы планировщика Go.
  - Почему планировщик Go такой, какой он есть.
- Каналы "под капотом"

# Зачем нужен планировщик?

# Зачем нужен планировщик?

Распределять вычислительные ресурсы между задачами.

# Зачем Go собственный планировщик?

- Уже есть планировщик процессов и потоков в ядре.
- Почему бы не запускать каждую горутину в отдельном потоке?



# Зачем Go собственный планировщик?

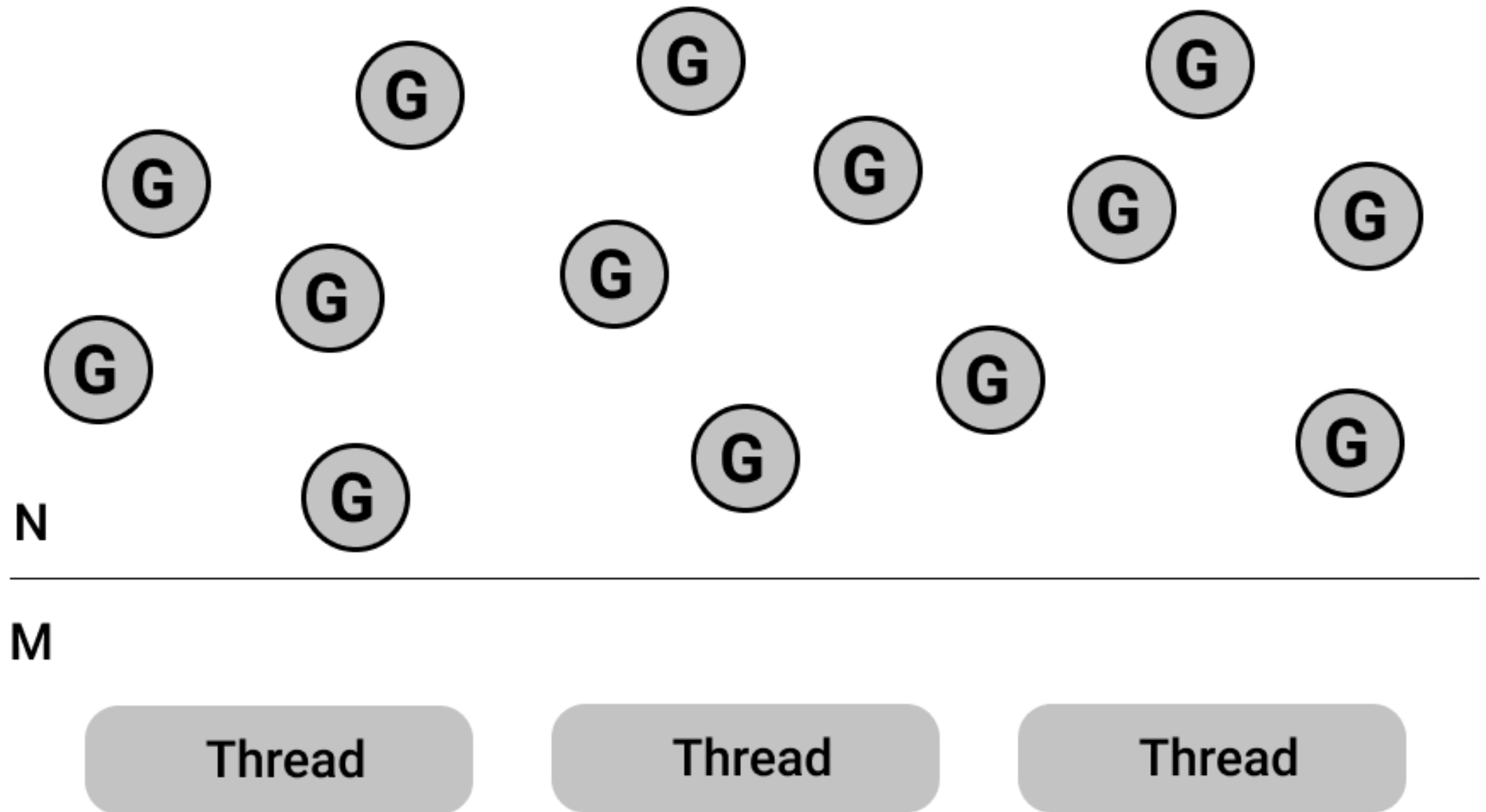
## Проблемы потоков

- Потоки дорогие по памяти (из-за стэка).
- Потоки дорого переключать (сисколы, контекст).

## Решения в Go

- Go использует growable stack.
- Go выбирает моменты, когда переключение дешевое.

# Проектируем планировщик: m-n threading

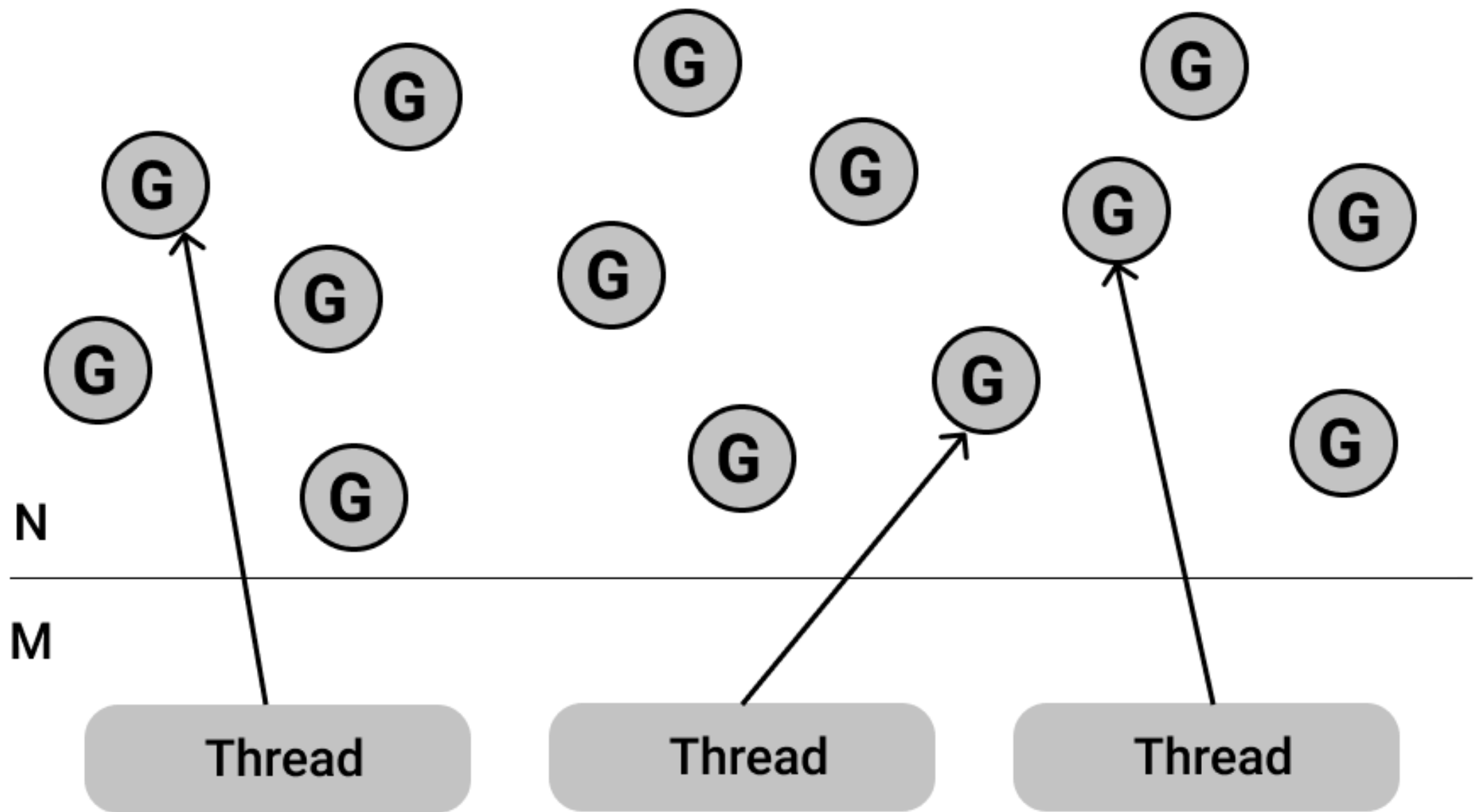


# Термины

- G (Goroutine): Легковесные потоки, управляемые Go, которые выполняют функции.
  - N - множество всех горутин
- M (Machine): OS thread. Поток Операционной системы под управлением средой выполнения Go.
- P (Processor): Логические процессоры, которые выполняют G на M.
  - GOMAXPROCS - это про P.
  - В контейнерах: <https://github.com/uber-go/automaxprocs>

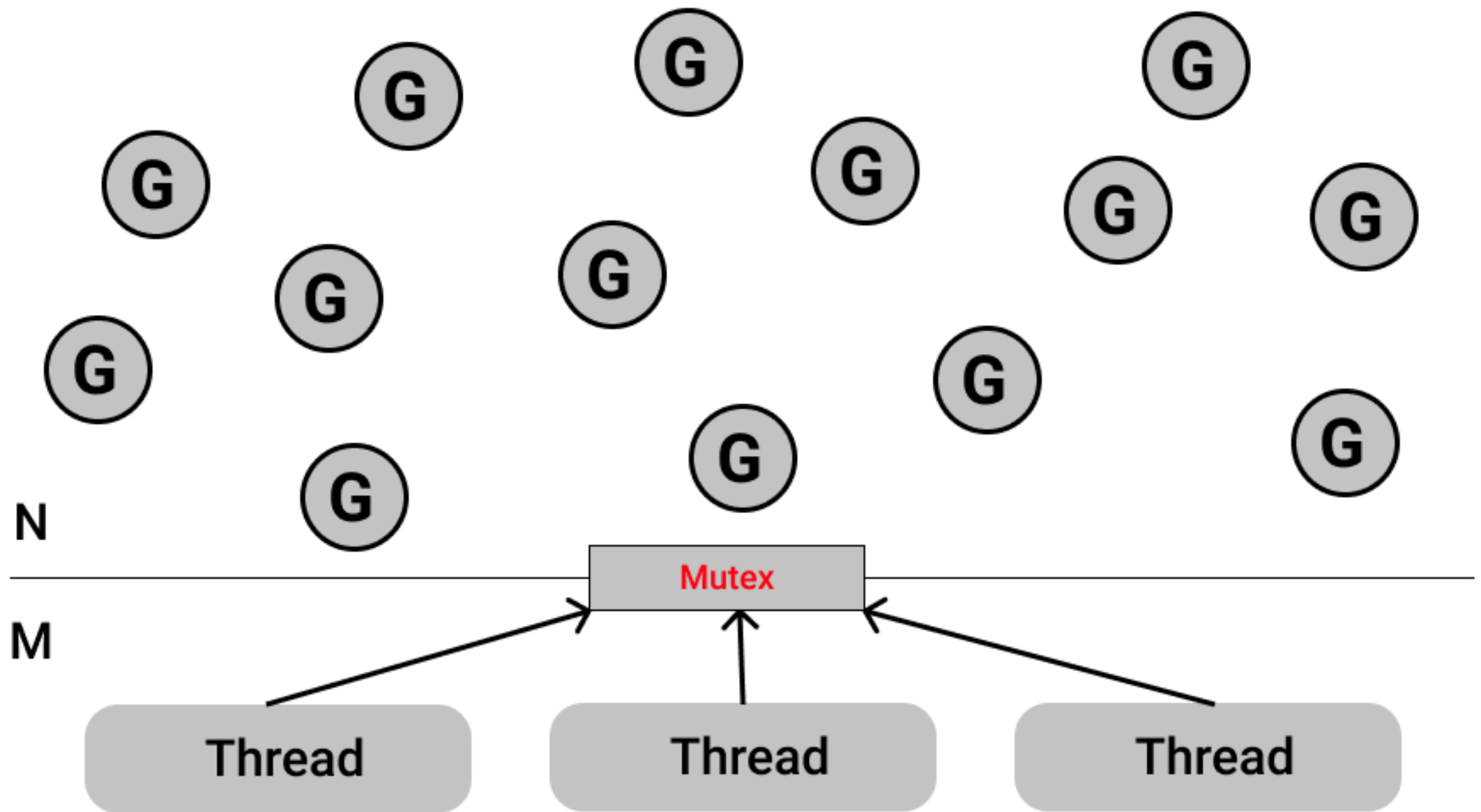
<https://go.dev/src/runtime/HACKING>

# Проектируем планировщик: m-n threading

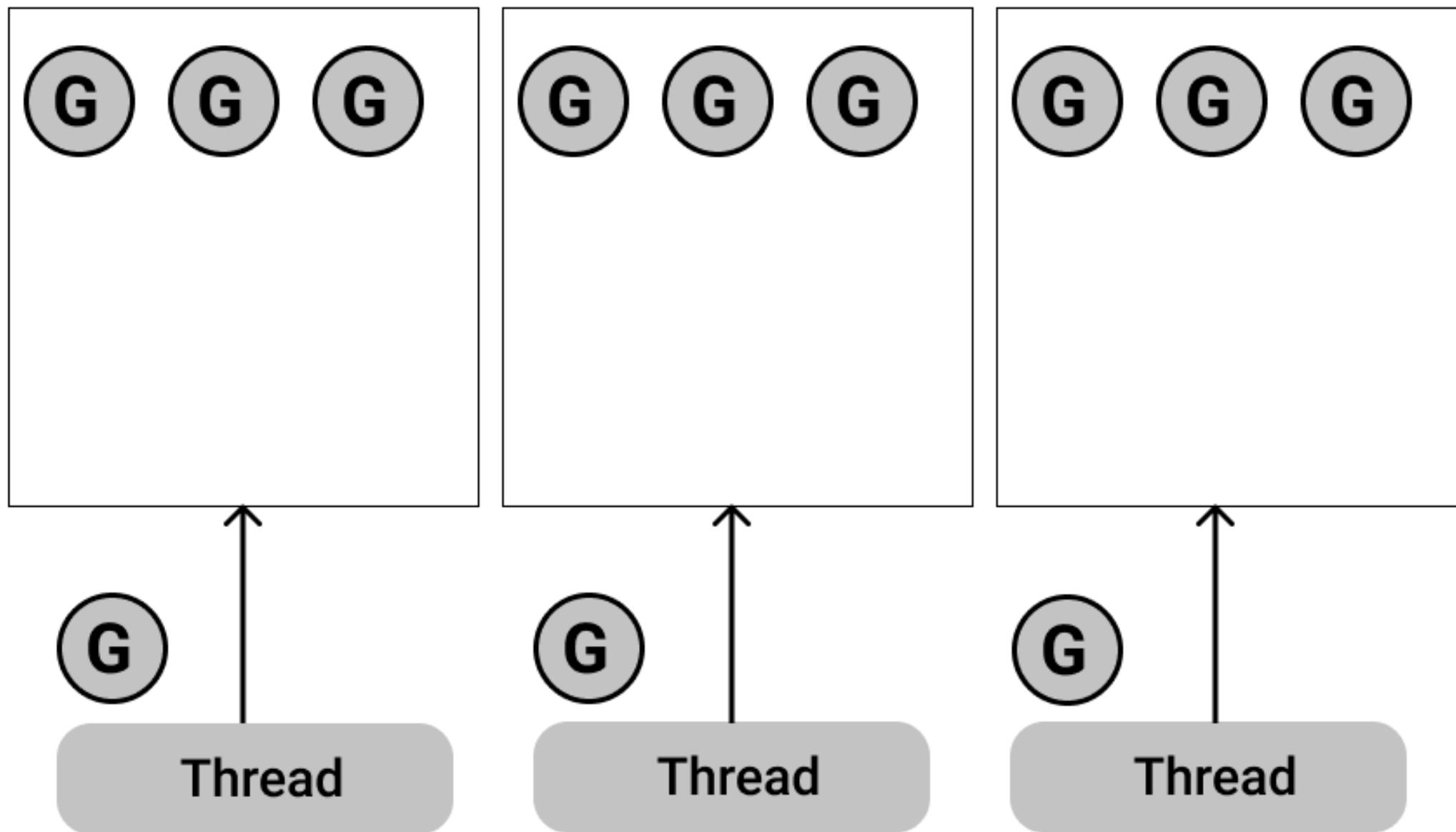


Какие есть проблемы?

# Проектируем планировщик: m-n threading



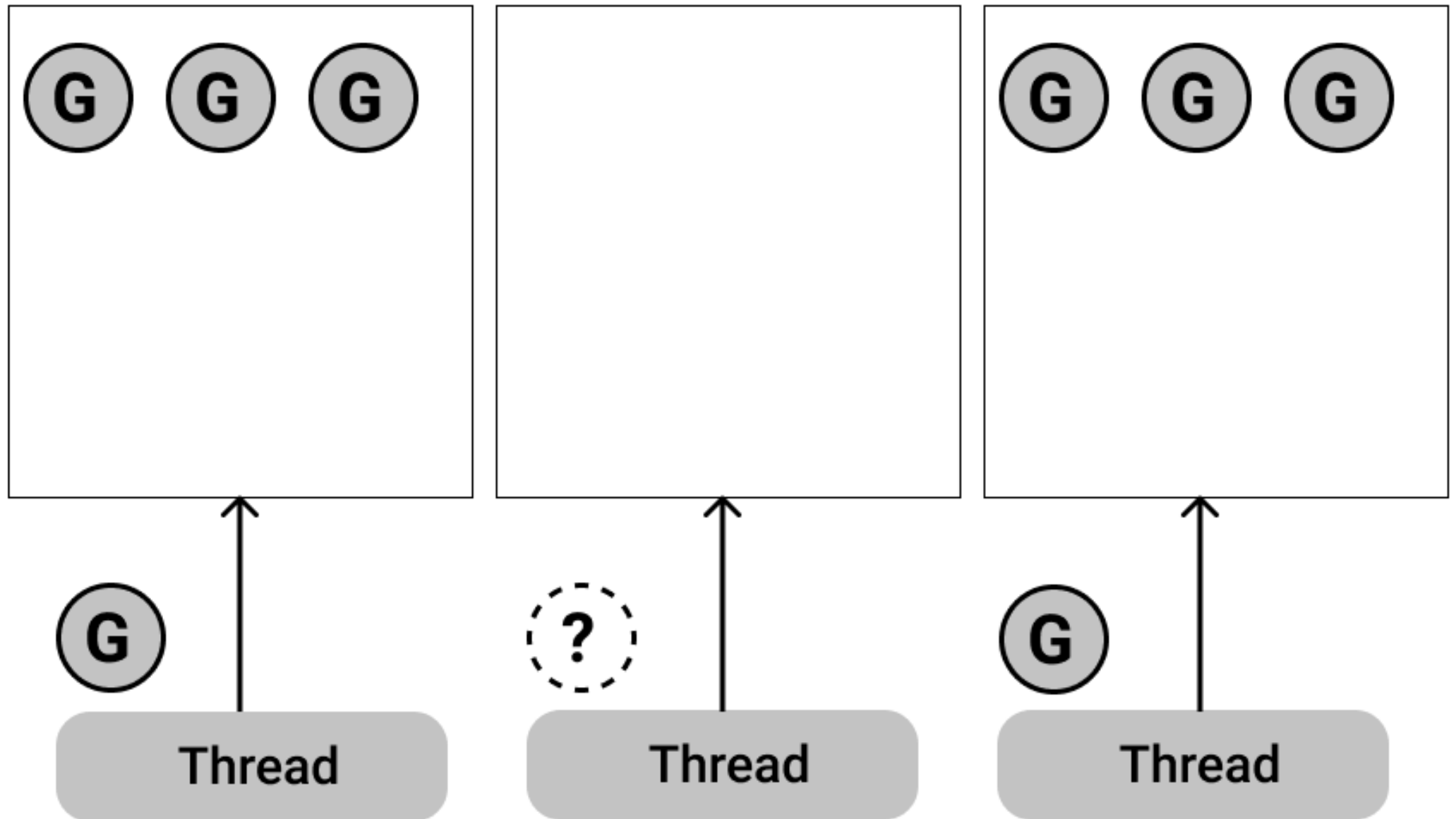
## Проектируем планировщик: отдельные очереди



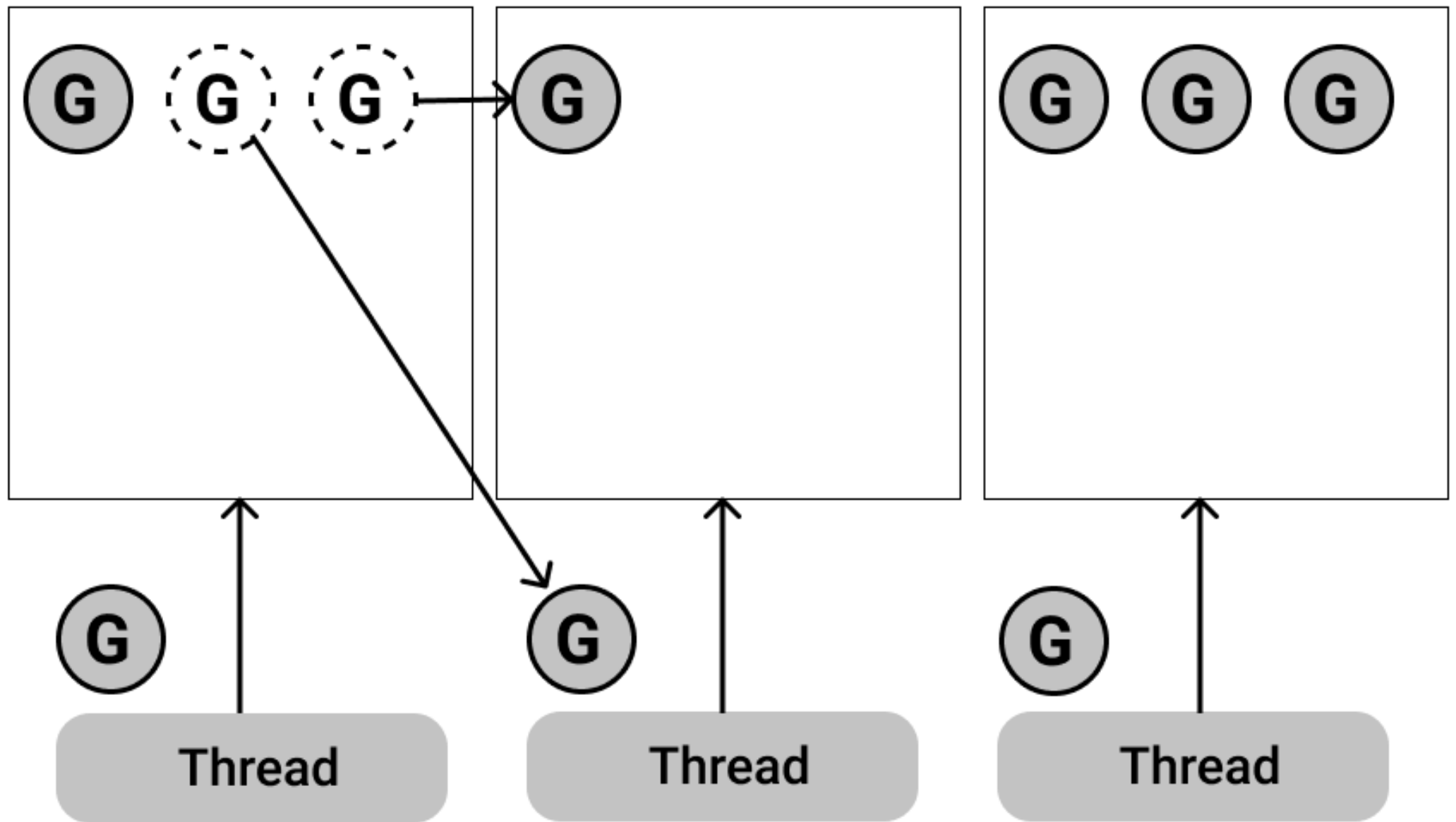
Какие есть проблемы?



# Проектируем планировщик: закончилась очередь

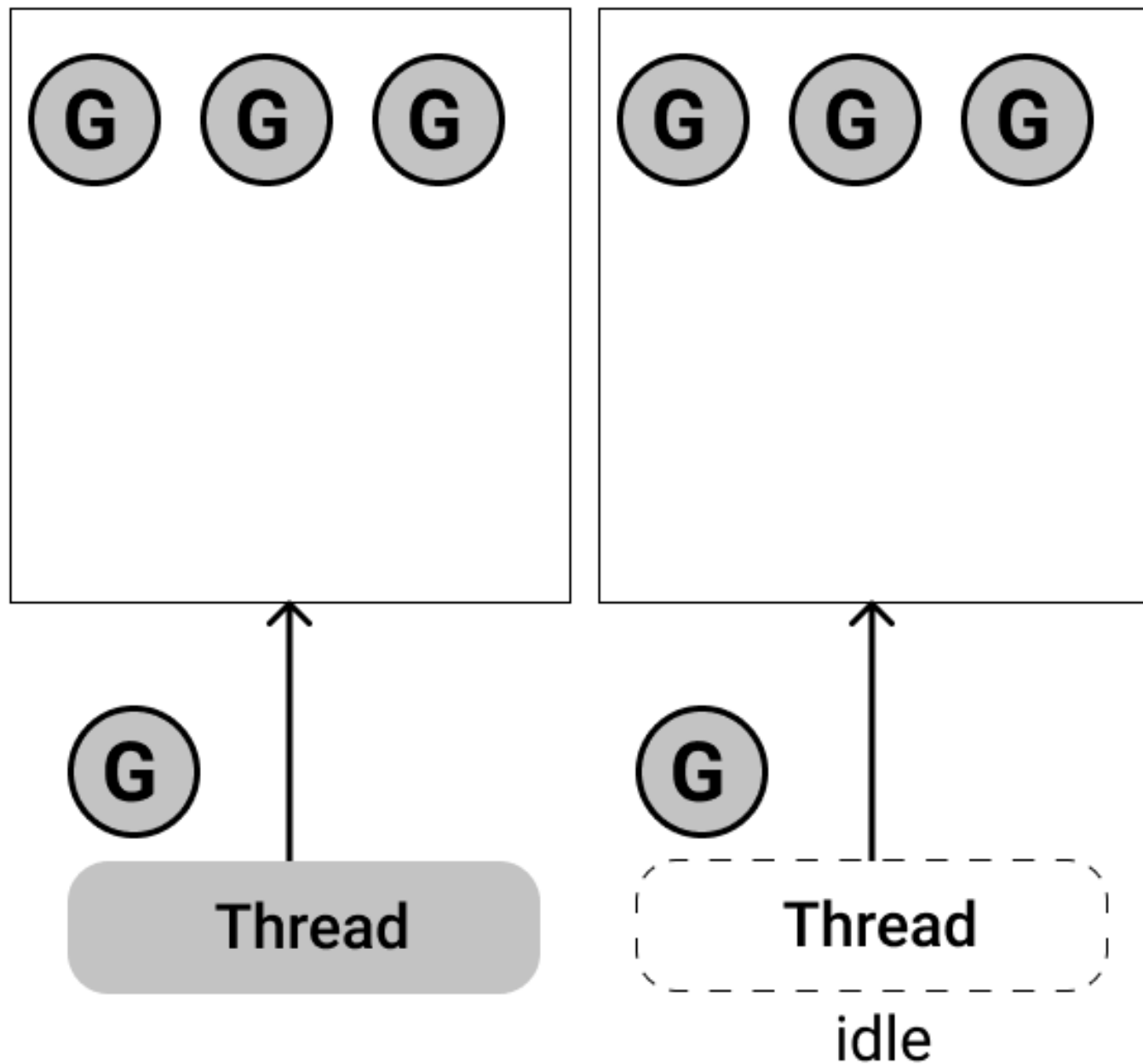


## Проектируем планировщик: work stealing



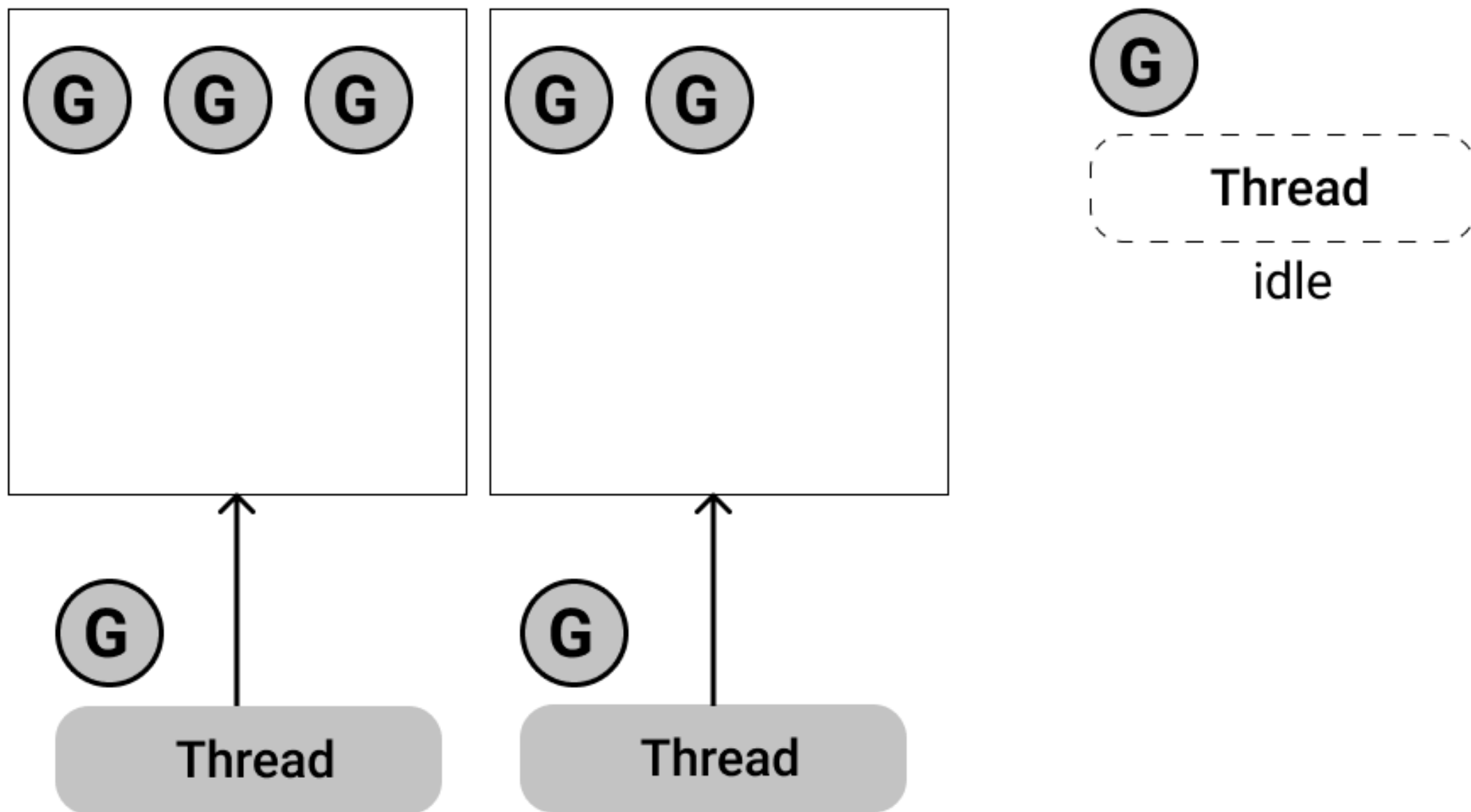


# Проектируем планировщик: syscall



Тред заблокирован.

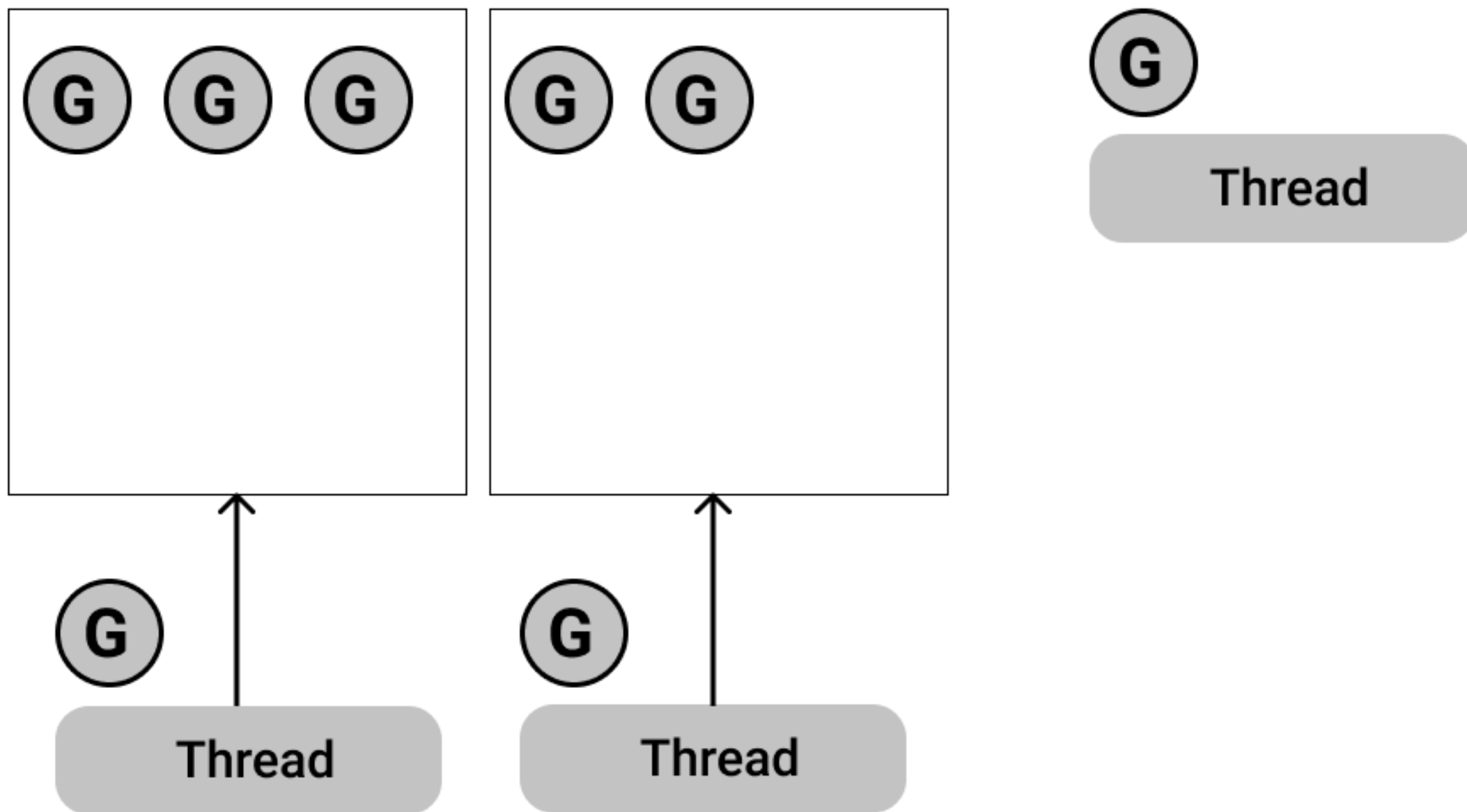
# Проектируем планировщик: syscall



Создаем новый тред.



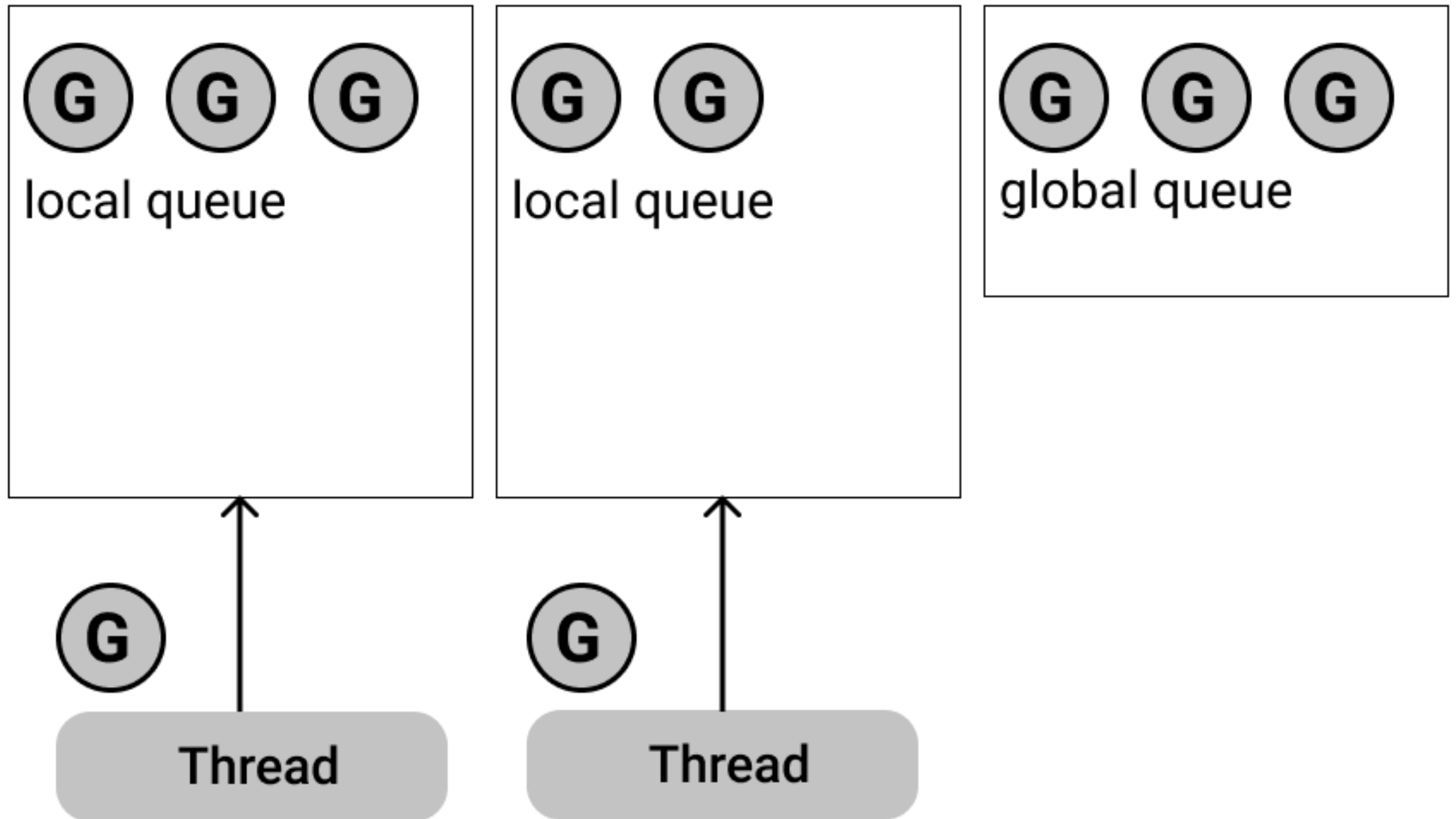
# Проектируем планировщик: syscall



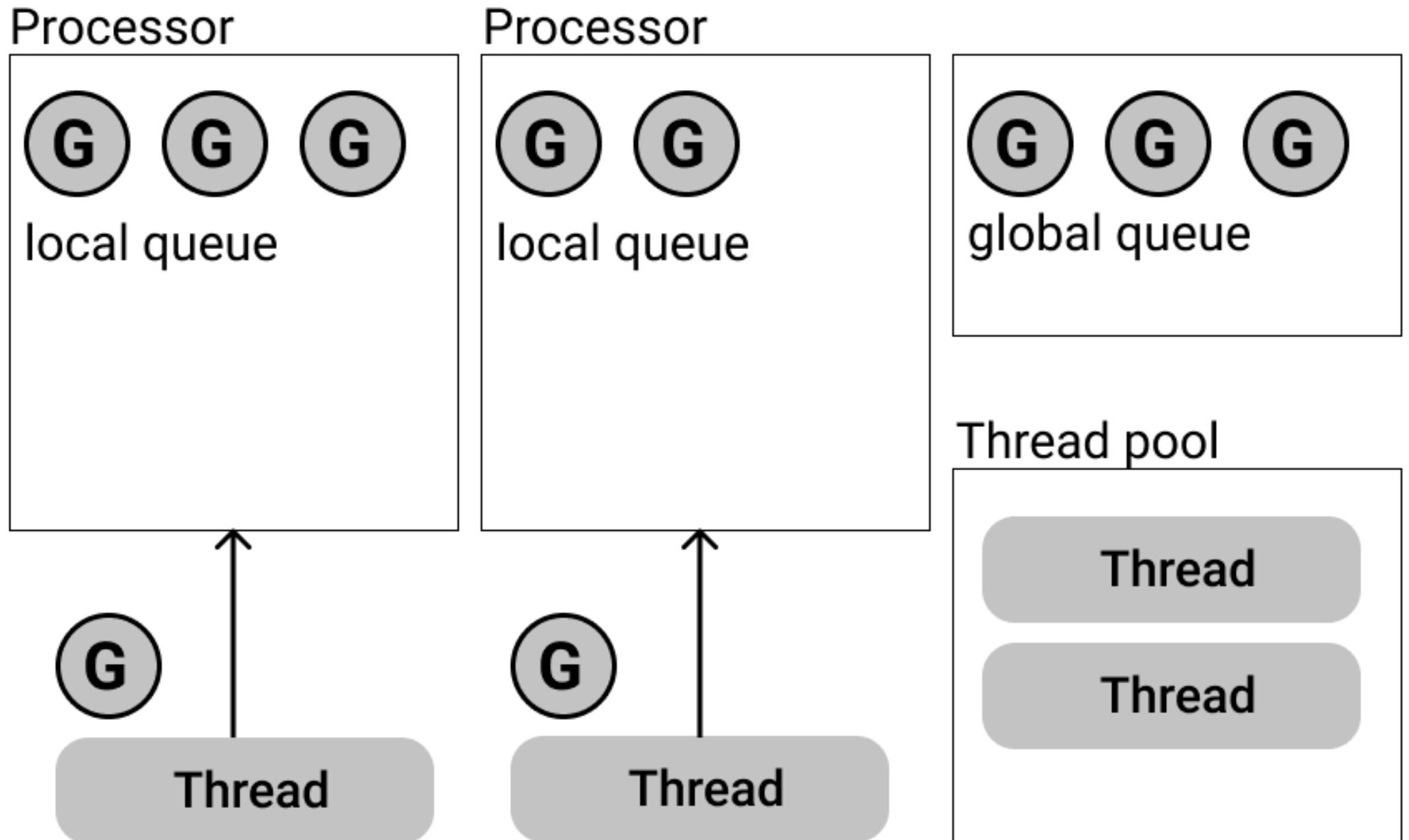
Куда деть горутину после syscall'a?



## Проектируем планировщик: глобальная очередь



# Проектируем планировщик: все идеи вместе



# Планировщик: go tool trace

- <https://golang.org/cmd/trace/>
- [Как мы оптимизировали наш DNS-сервер с помощью инструментов GO](#)
- <https://blog.gopheracademy.com/advent-2017/go-execution-tracer/>

# Планировщик: честность

FIFO Run Queue



LIFO Run Queue



Какой тип очереди честнее?



# Планировщик: честность

**Fair: FIFO Run Queue**



**Not Fair: LIFO Run Queue**



Какие проблемы?

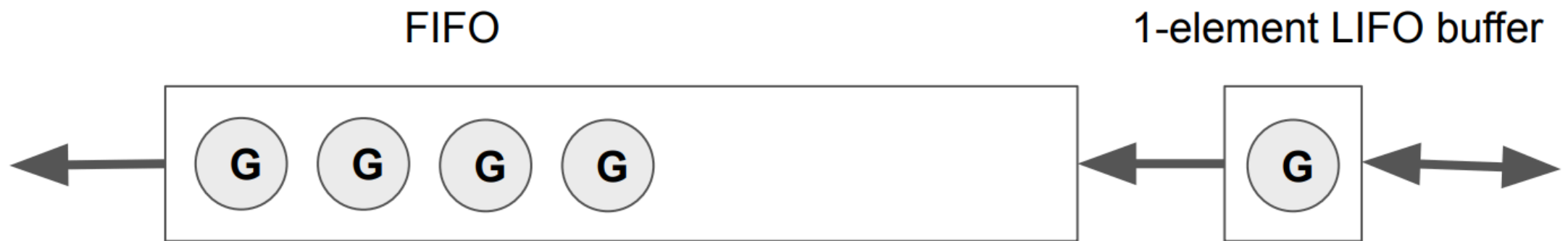




# Планировщик: честность, trade-off

За честность приходится платить производительностью.

# Планировщик: честность, trade-off



Одноэлементное LIFO улучшает использование кэша -> дешевле переключать горютины.



# Планировщик: голодание FIFO

- А что если две горутины постоянно ставят друг друга в LIFO?

.

# Планировщик: голодание FIFO

- А что если две горутины постоянно ставят друг друга в LIFO?

Считать время непрерывной работы цепочки горутин.

# Планировщик: голодание горутины

- А что если одна горутина находится в бесконечном цикле?

# Планировщик: голодание горутины

Go 1.14 Asynchronous Preemption:

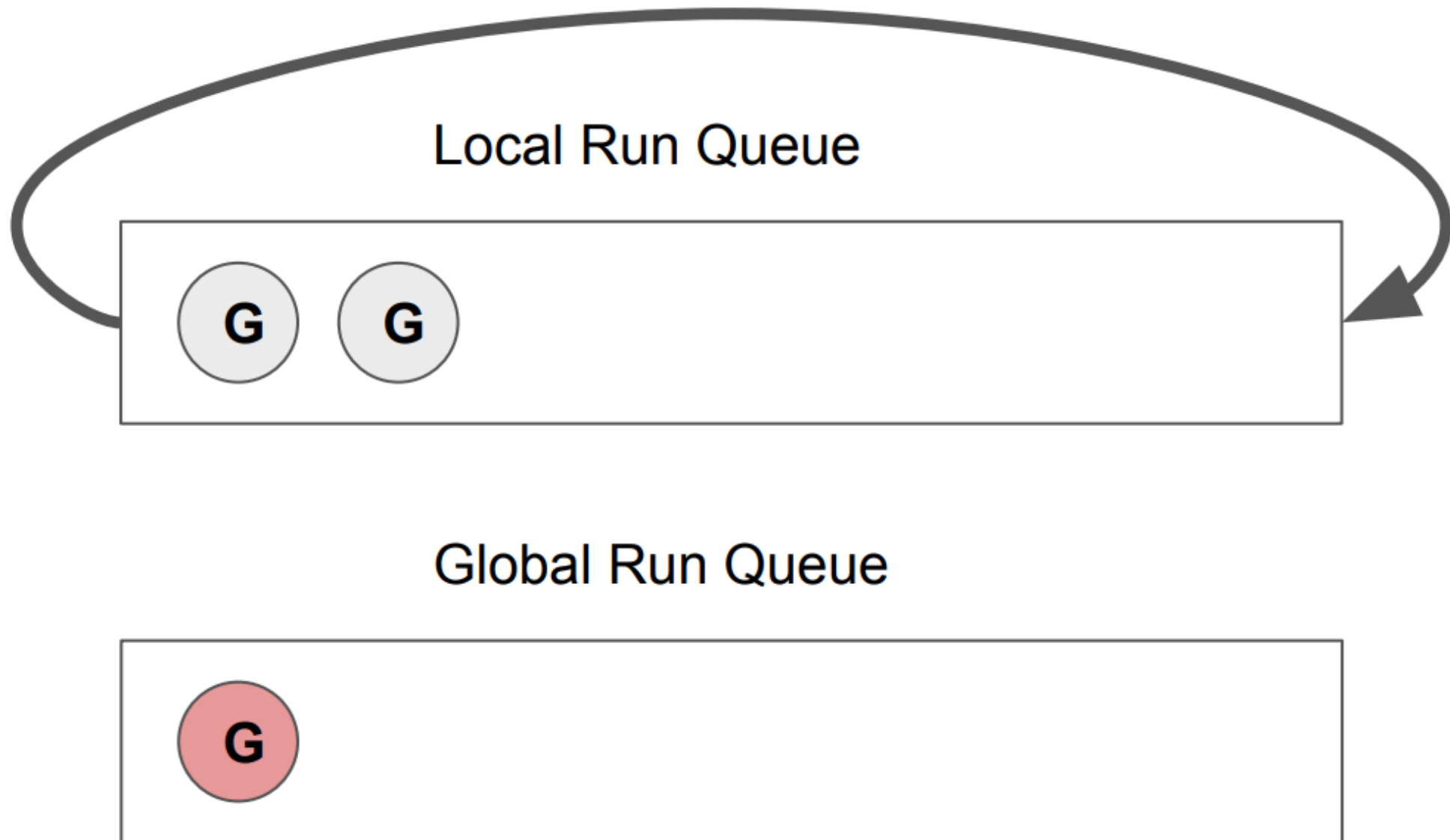
- <https://medium.com/a-journey-with-go/go-asynchronous-preemption-b5194227371c>
- <https://github.com/golang/proposal/blob/master/design/24543-non-cooperative-preemption.md>

# Планировщик: порядок поиска работы

- Локальная очередь
- Глобальная очередь
- Work stealing

# Планировщик: голодание глобальной очереди

- А что если горютины в локальных очередях не кончаются?





# Планировщик: голодание глобальной очереди

- А что если горютины в локальных очередях не кончаются?

Брать горютины из глобальной очереди каждый 61-й тик.

# Планировщик: network poller

Проверка готовности горютины в очереди — это проверка значения в мапе.

- Это быстро и дешево.

Проверка готовности сетевого io — это syscall

- Syscall паркует тред.
- Syscall это дорого.

## Что делать?

## Планировщик: network poller

- Завести для network poller отдельный тред

# Планировщик: честность, саммари

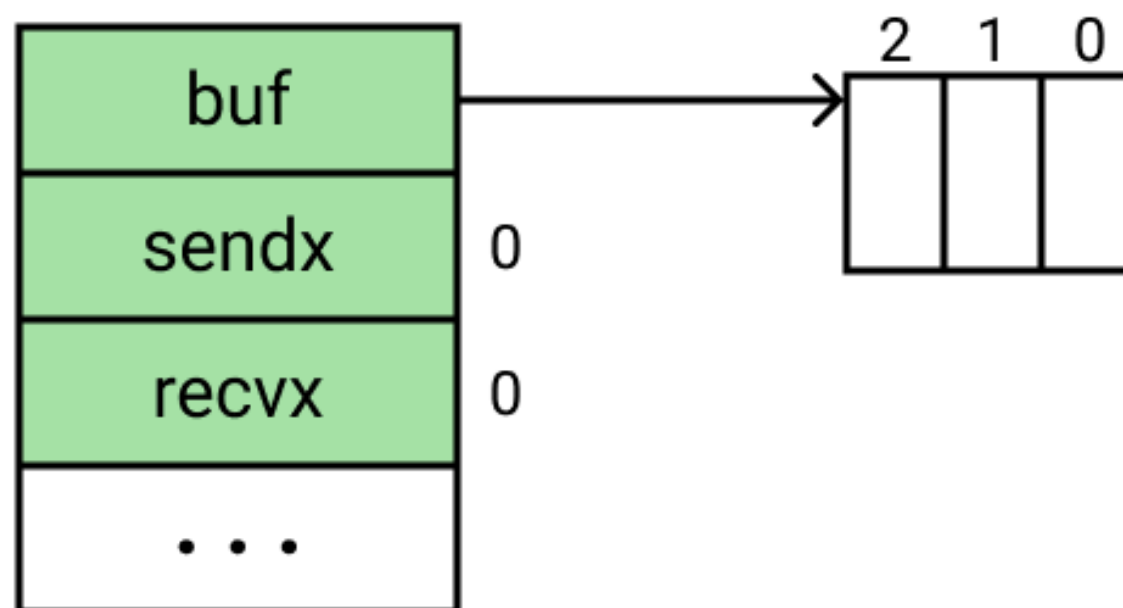
- Бесконечные горютины — sysmon
- Циклы из горютин — давать работать суммарно 10 мс
- Глобальная очередь — периодически брать горютины из нее
- Network poller — отдельный тред

# Каналы

<https://golang.org/src/runtime/chan.go>

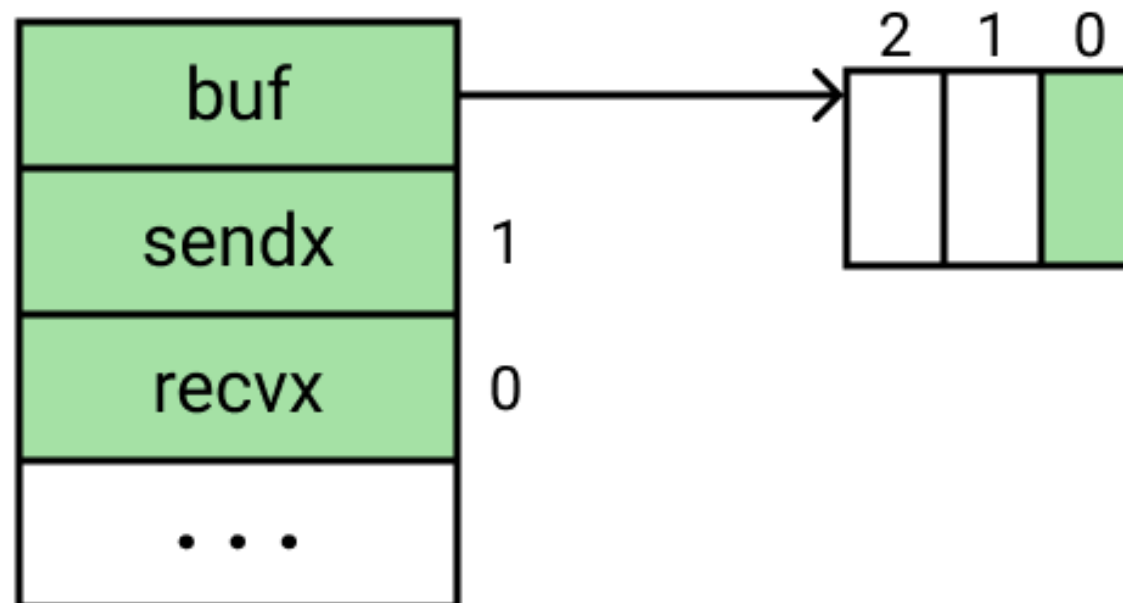
# Каналы

```
ch := make(chan string, 3)
```



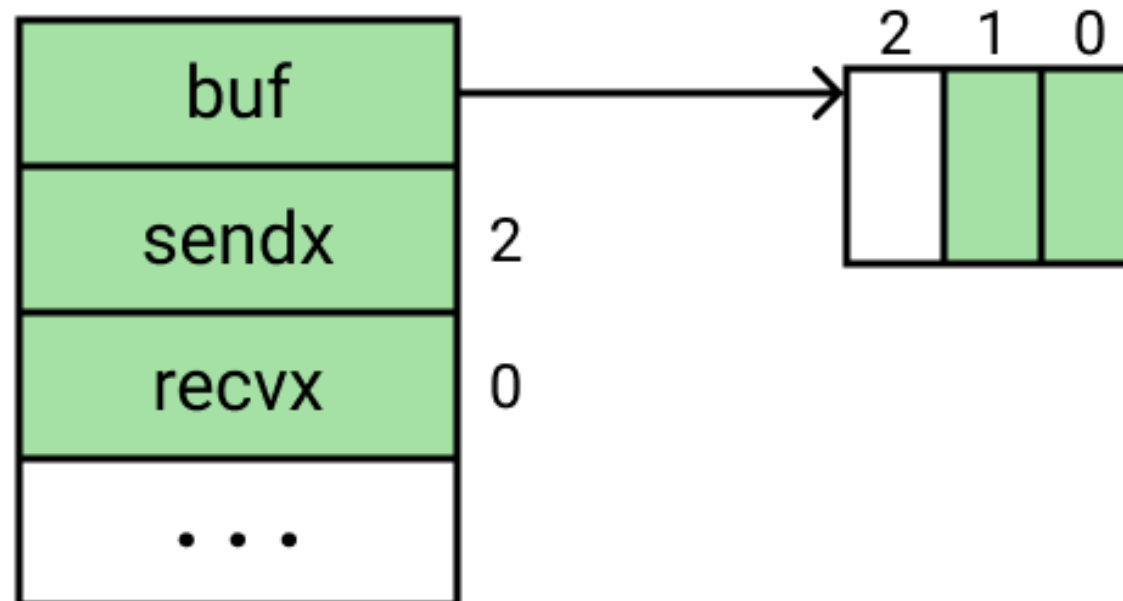
# Каналы: запись

ch <- "hello"



## Каналы: запись

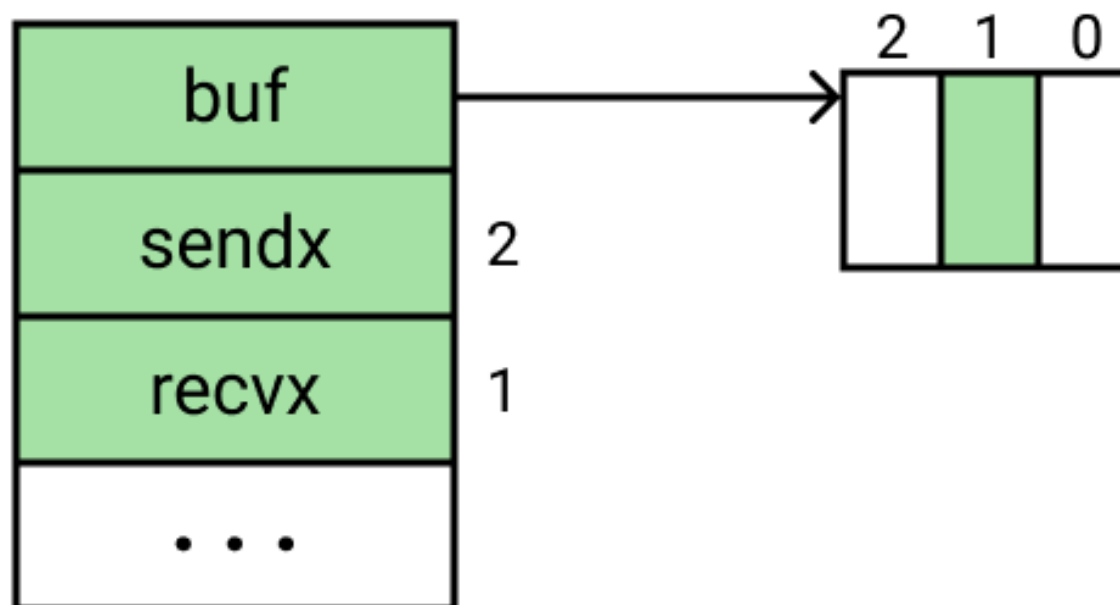
ch <- "world"



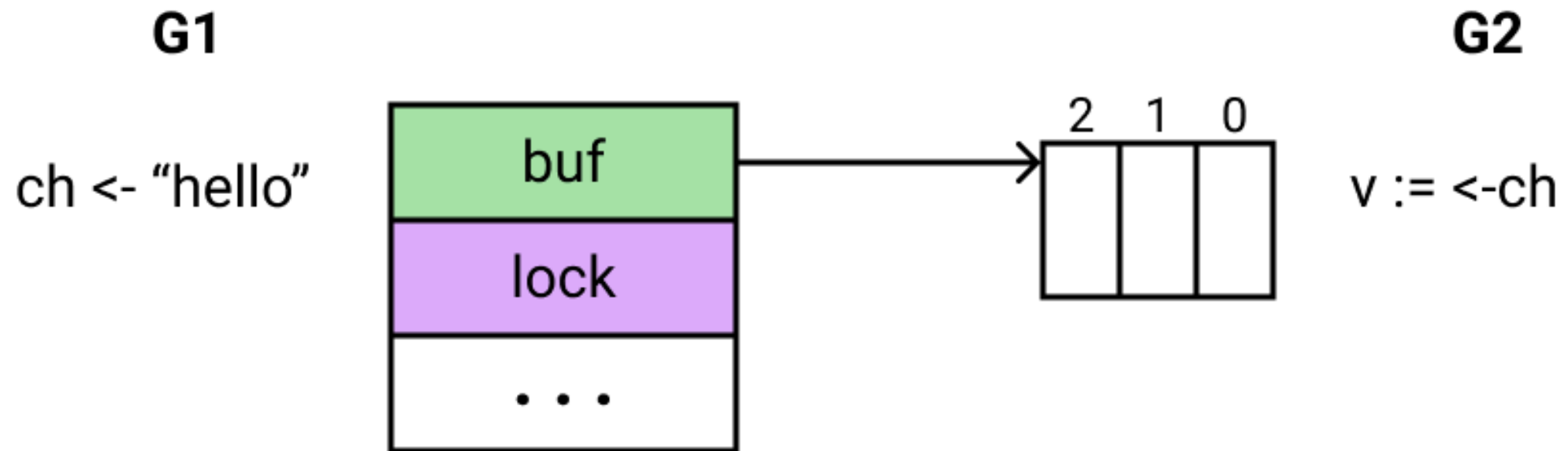


# Каналы: чтение

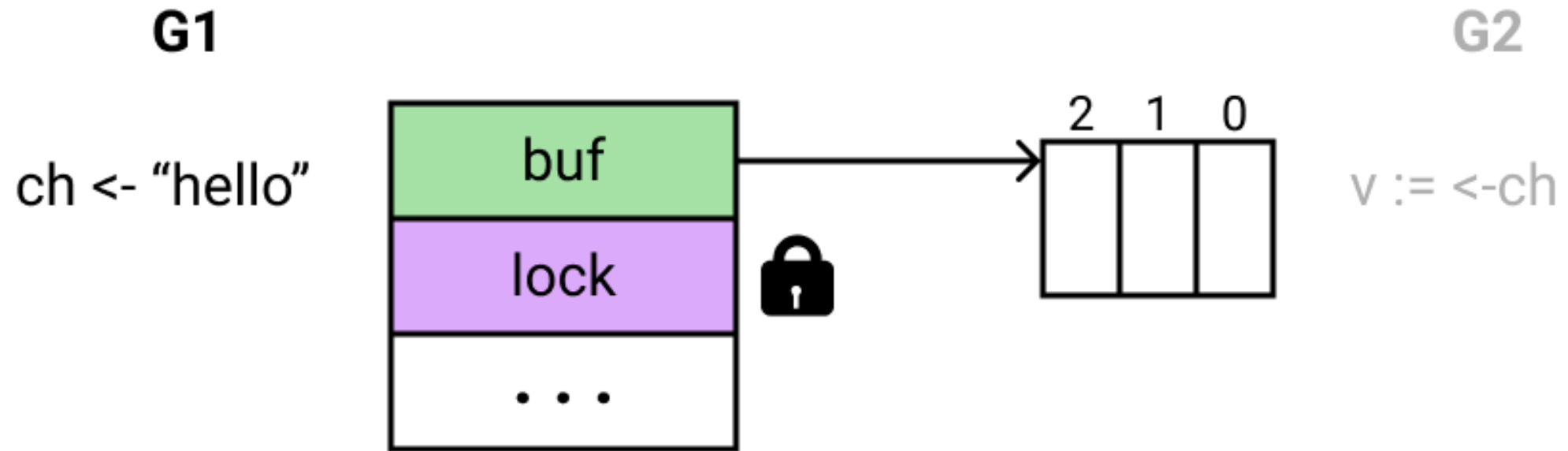
$v := \leftarrow ch$



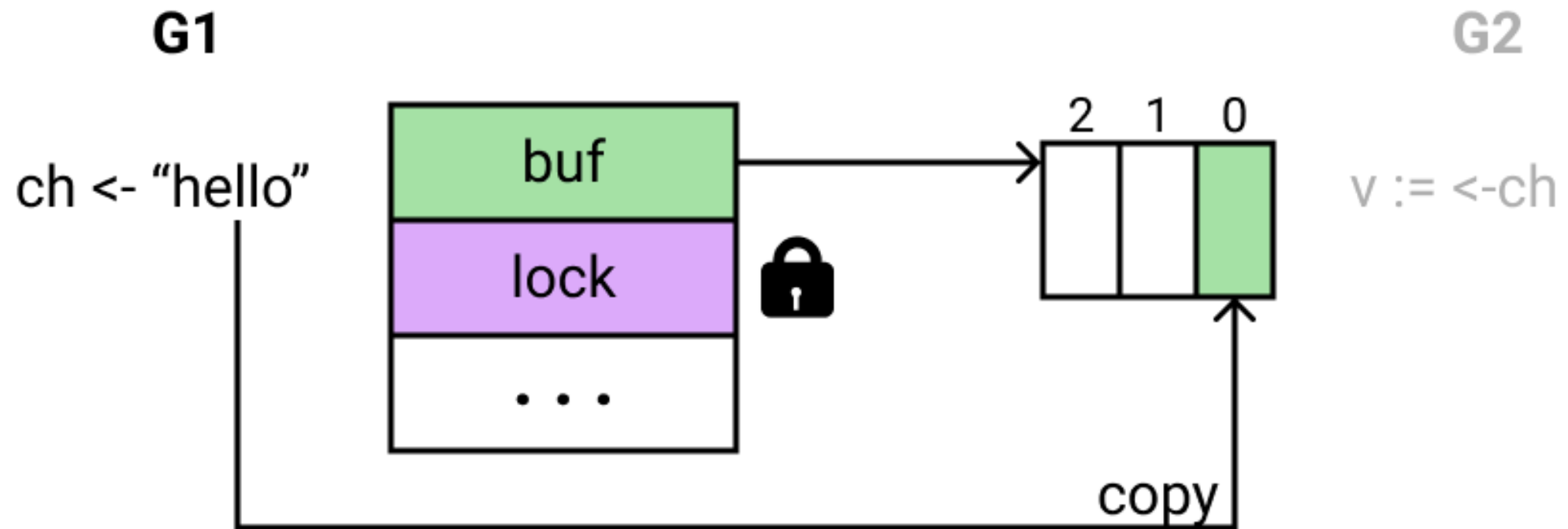
# Каналы: concurrency



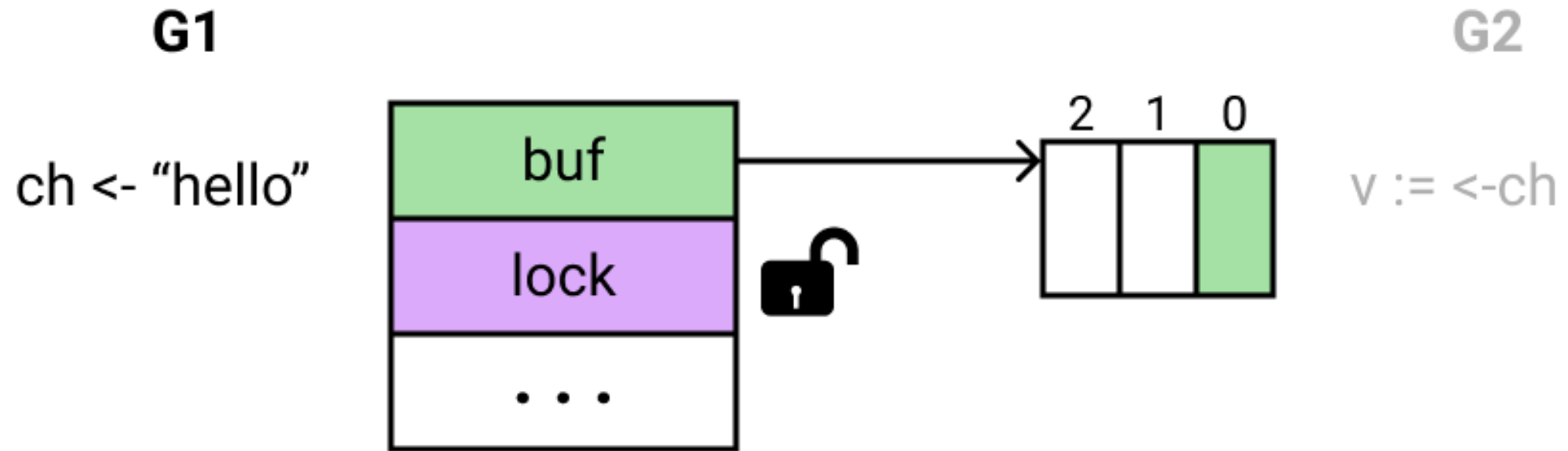
# Каналы: concurrency



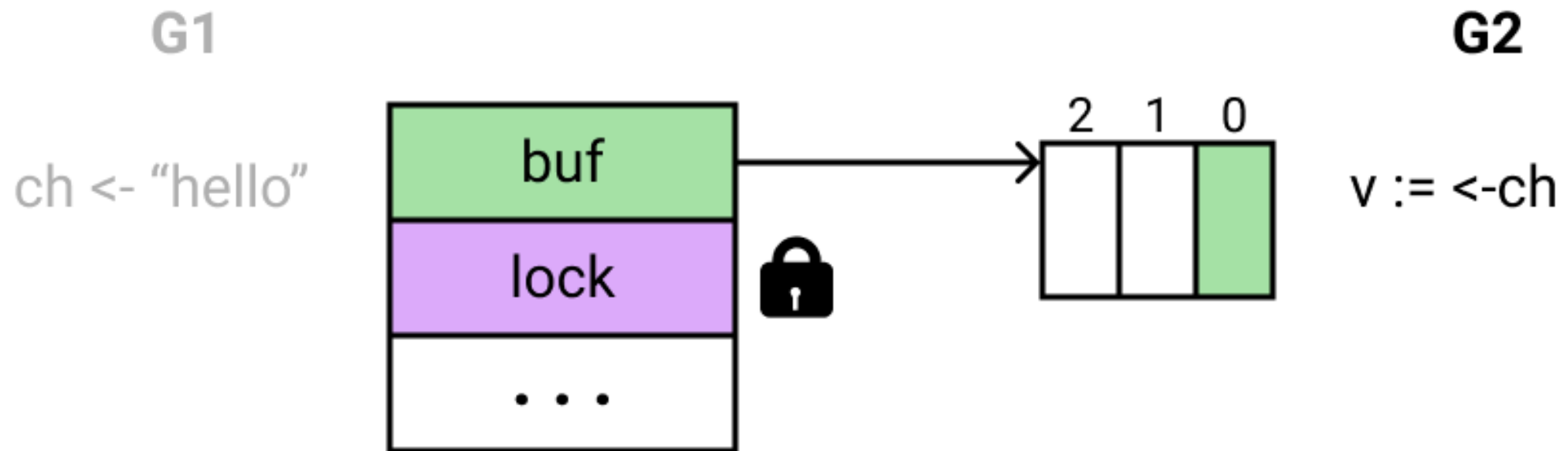
# Каналы: concurrency



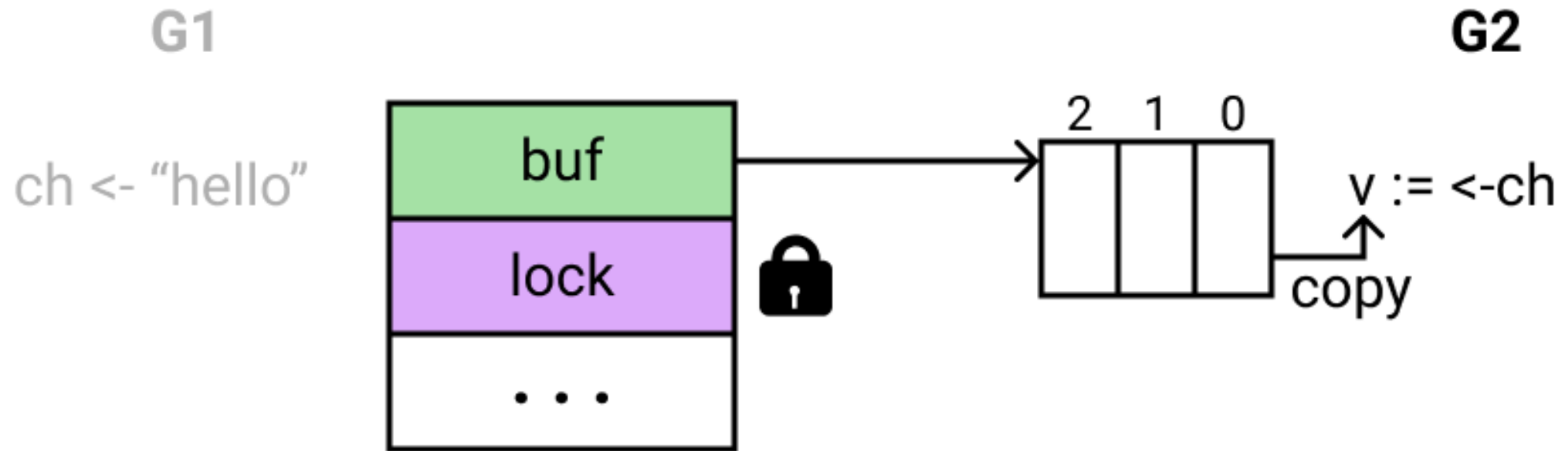
# Каналы: concurrency



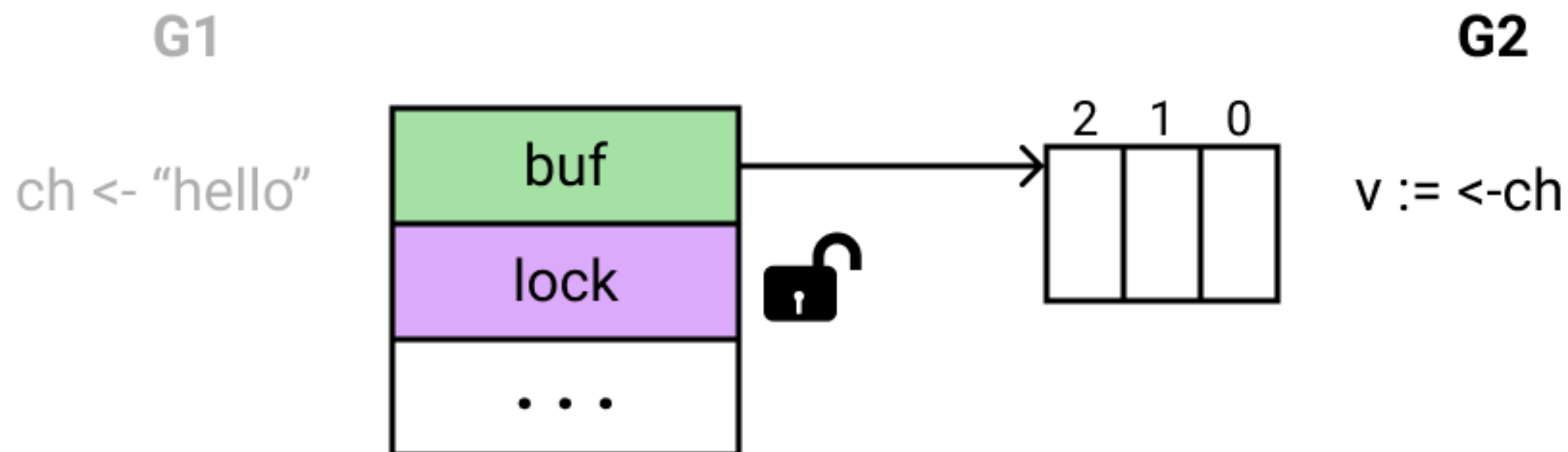
# Каналы: concurrency



# Каналы: concurrency

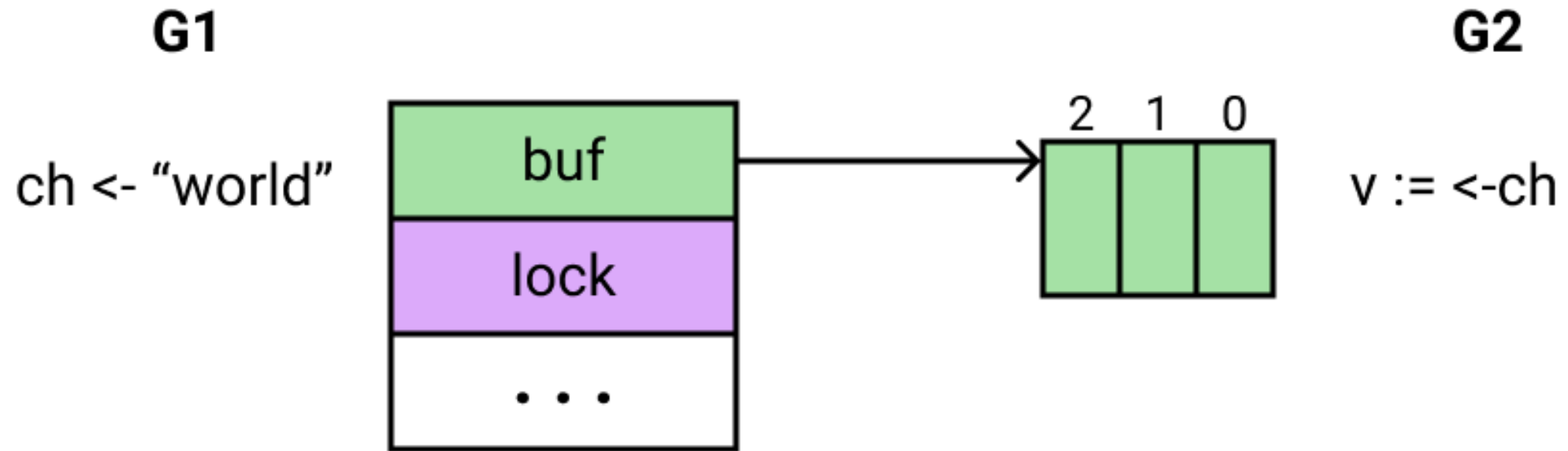


# Каналы: concurrency

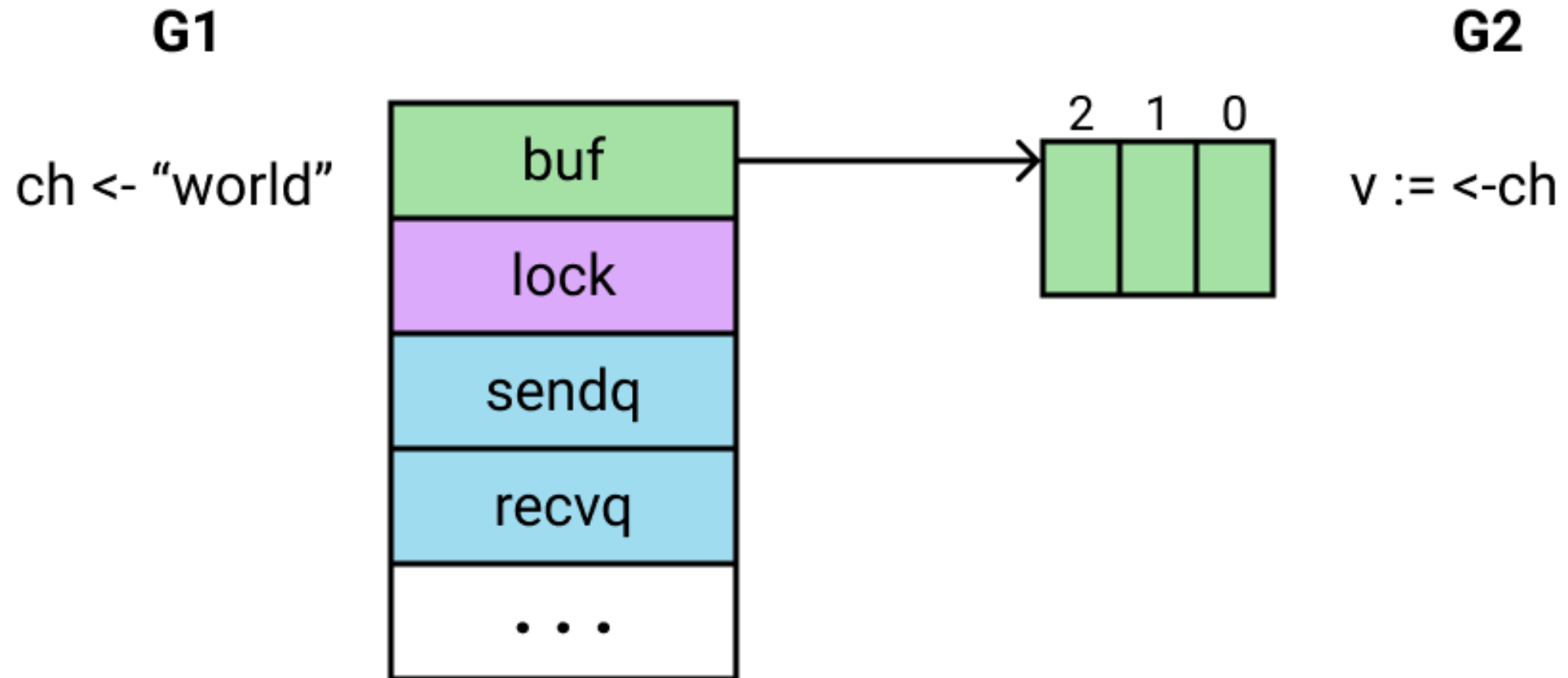




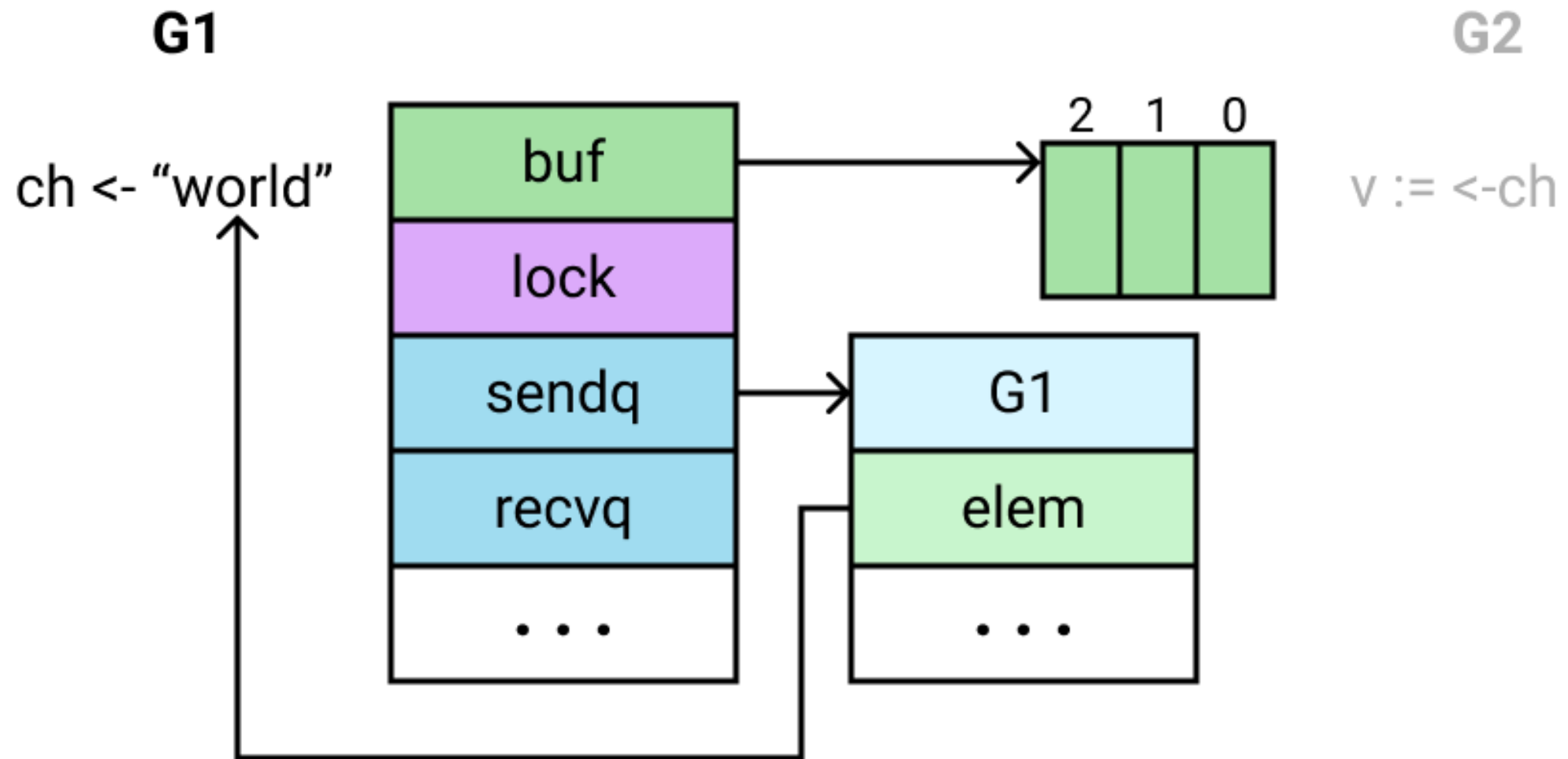
## Каналы: concurrency (write full)



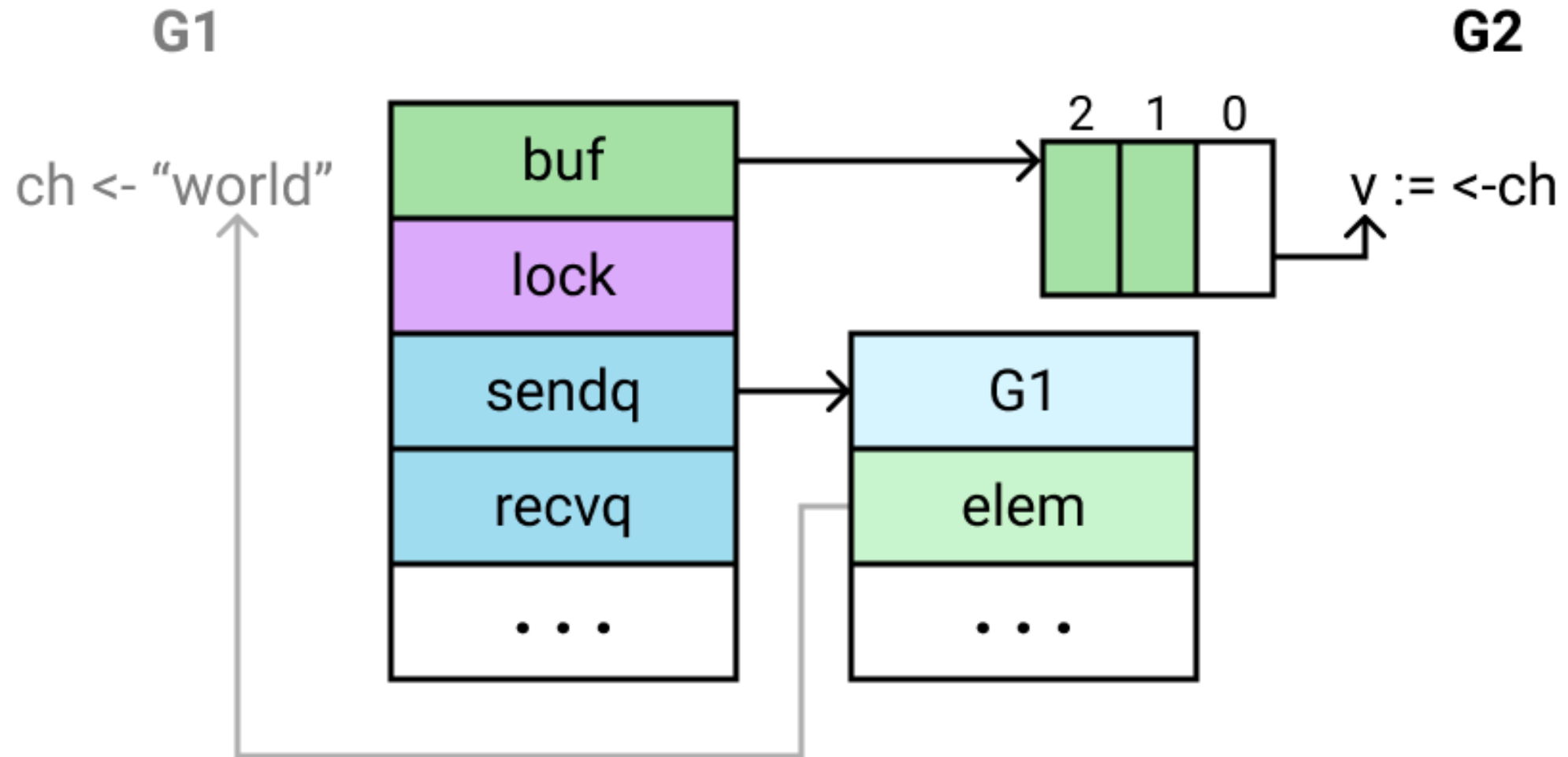
## Каналы: concurrency (write full)



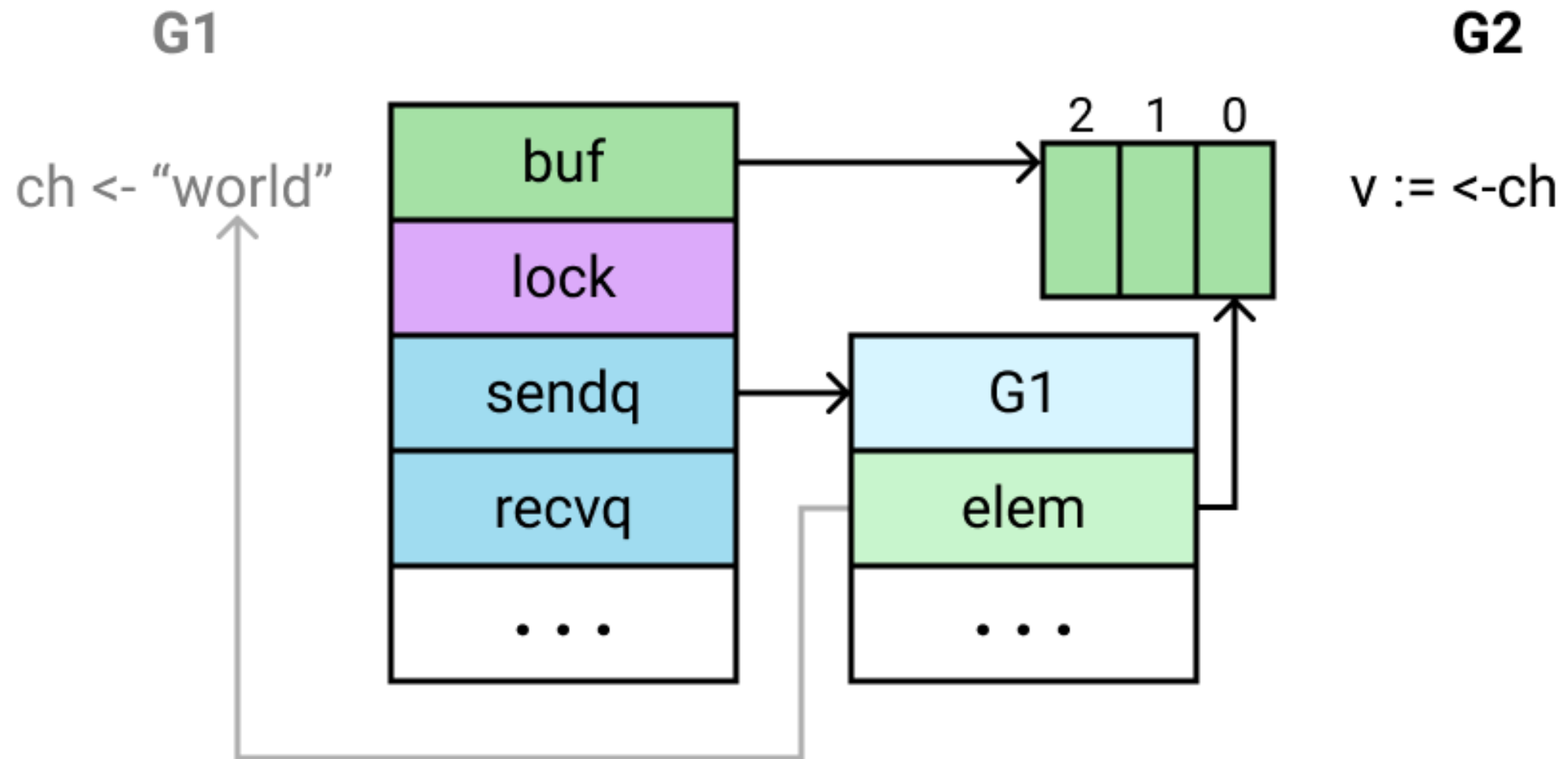
## Каналы: concurrency (write full)



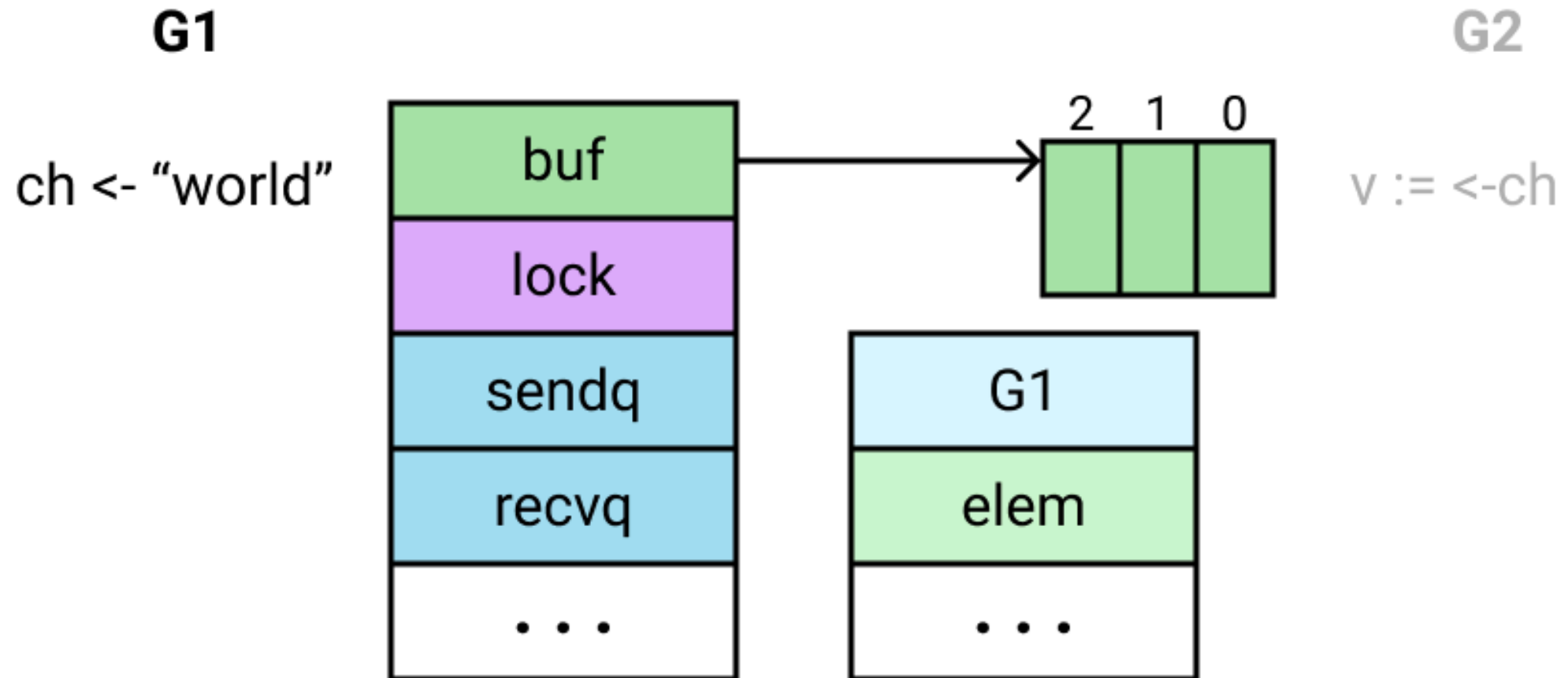
## Каналы: concurrency (write full)



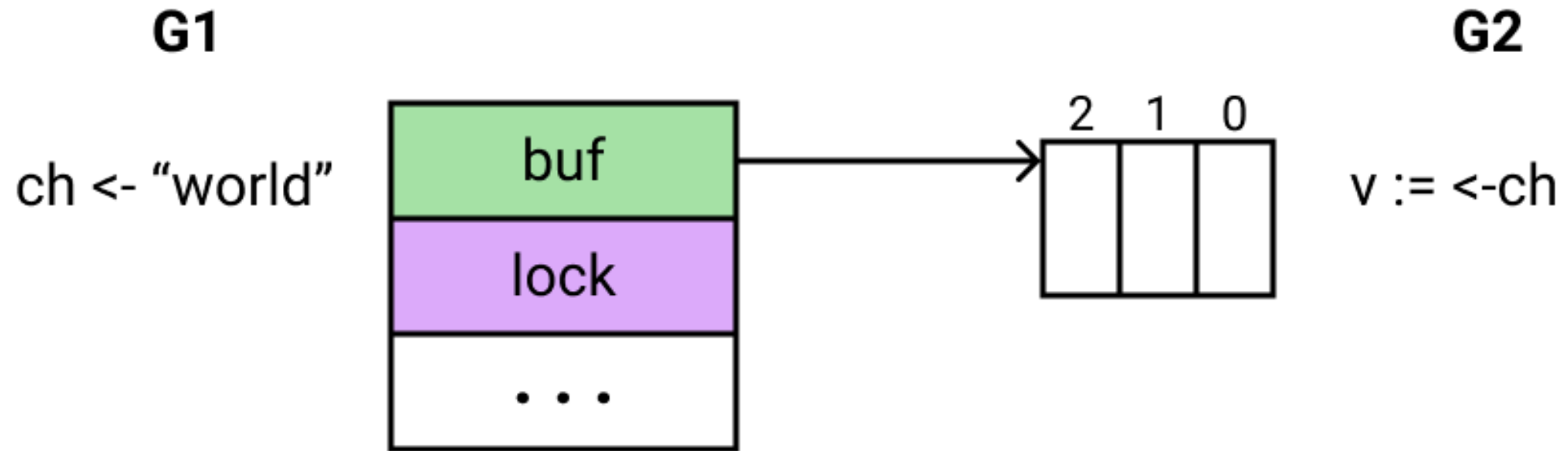
## Каналы: concurrency (write full)



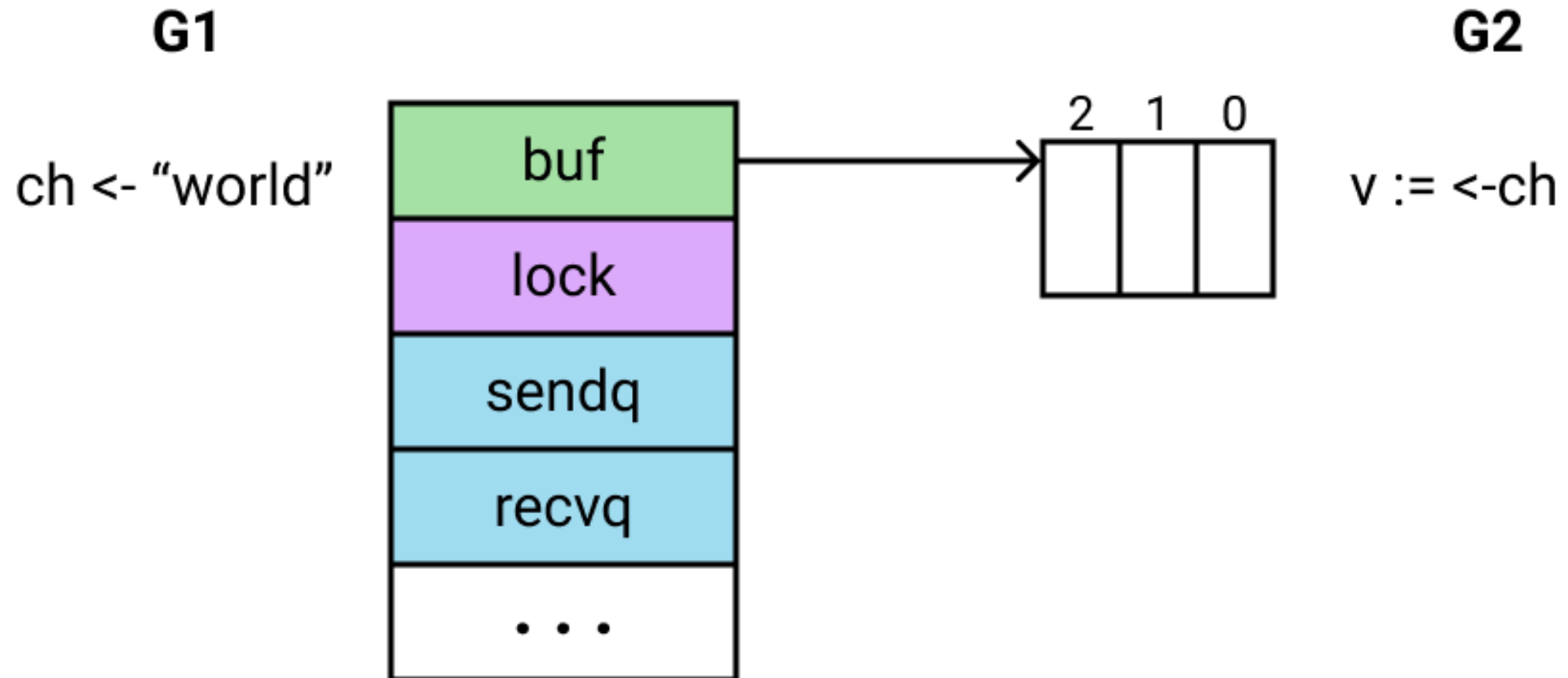
## Каналы: concurrency (write full)



## Каналы: concurrency (read empty)

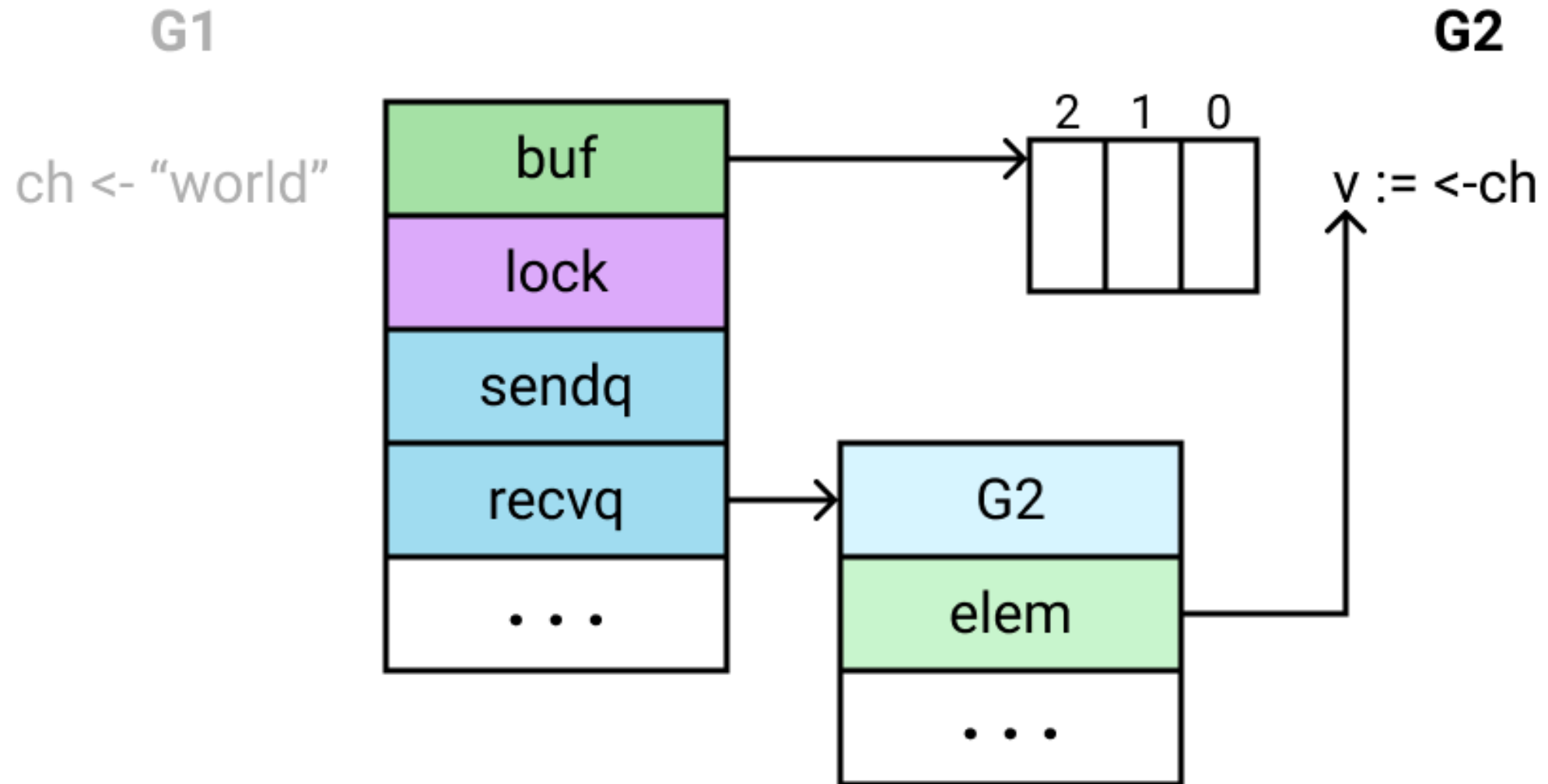


## Каналы: concurrency (read empty)

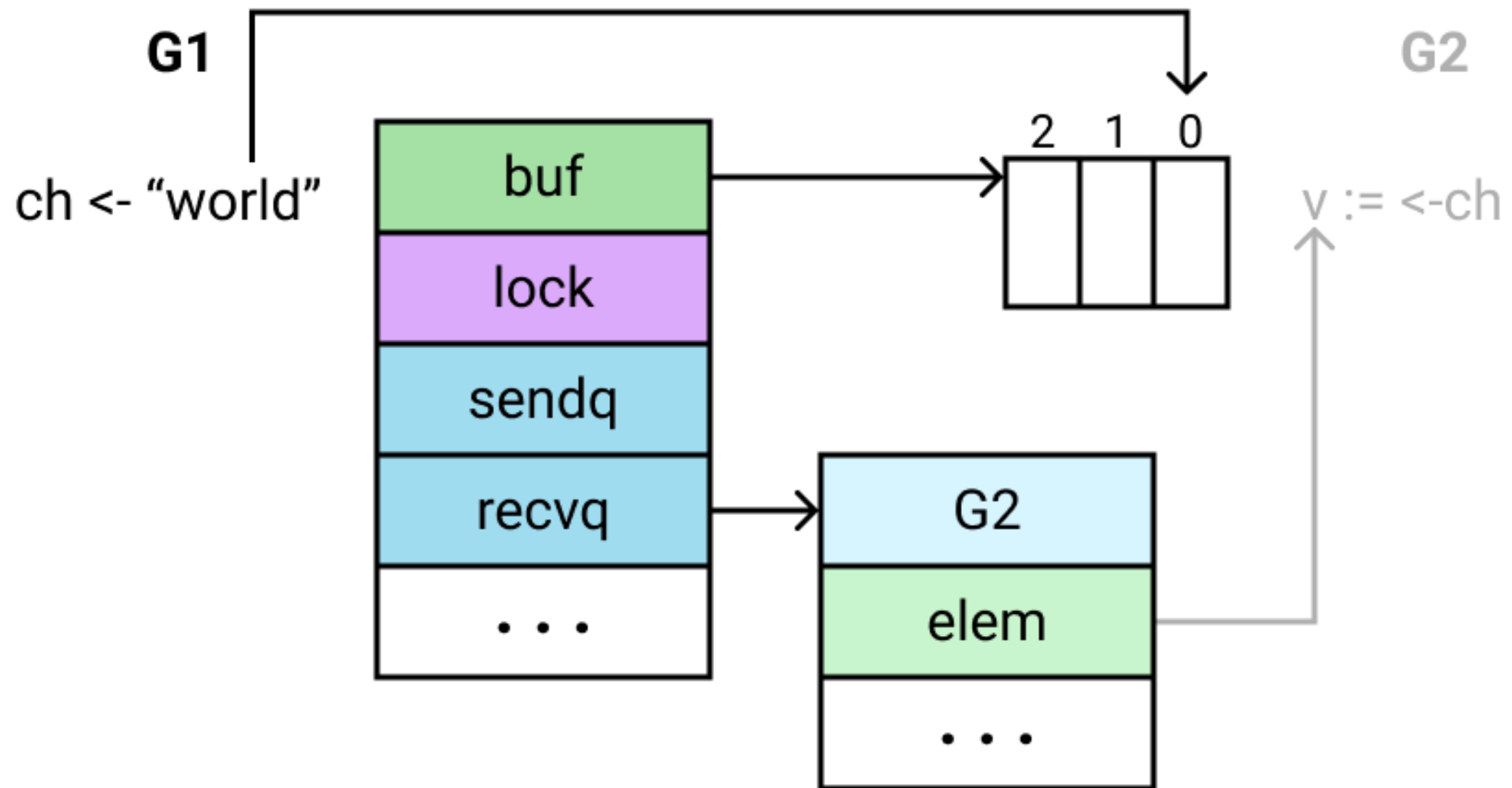




## Каналы: concurrency (read empty)



## Каналы: concurrency (read empty)

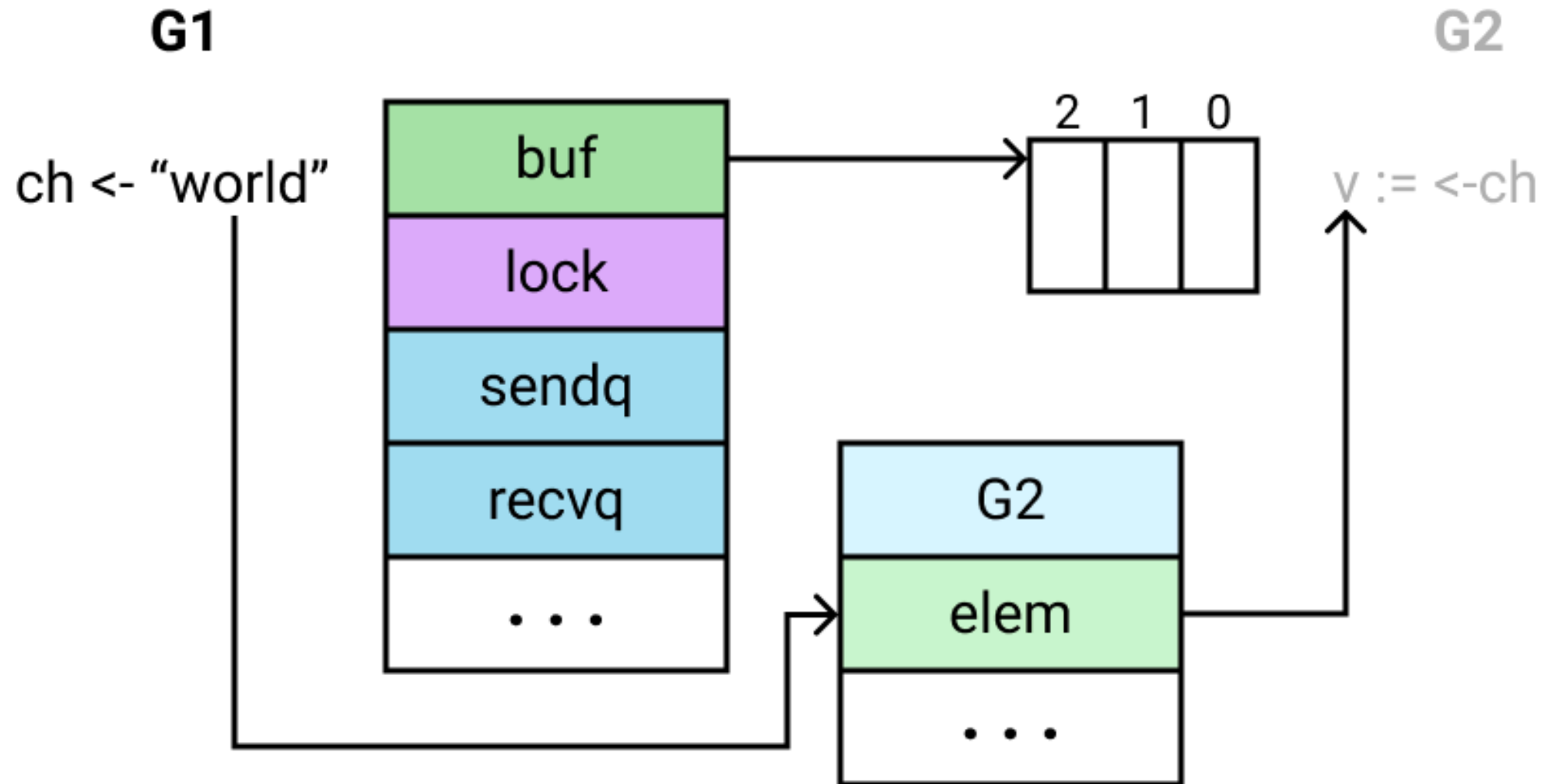


# НЕТ!

## ДВОЙНОМУ КОПИРОВАНИЮ



## Каналы: concurrency (read empty)



# Каналы: горутина пишет в стек другой горутины!

<https://cs.opensource.google/go/go/+/master:src/runtime/chan.go;l=211>

# Материалы

- Сага о планировщике — <https://youtu.be/YHR05WQGh0k>
- Планировщик шаг за шагом — <https://youtu.be/-K11rY57K7k>
- Про каналы — <https://www.youtube.com/watch?v=KBZIN0izeiY>
- Про планировщик на русском — <https://youtu.be/Gy6XEYWYht8>

# Вопросы?



Ставим “+”,  
если вопросы есть



Ставим “-”,  
если вопросов нет

**Заполните, пожалуйста,  
опрос о занятии  
по ссылке в чате**



# Следующий вебинар

19 марта

Go внутри. Память и сборка мусора



Ссылка на вебинар будет в ЛК за 15 минут



Материалы к занятию в ЛК — можно изучать



Обязательный материал обозначен красной лентой



Спасибо за внимание!

# Приходите на следующие вебинары

Алексей Романовский

