Планировщик процессов для гетерогенных процессорных архитектур

Шаго Павел Евгеньевич

Научный руководитель: Корхов Владимир Владиславович

Факультет прикладной математики – процессов управления
Кафедра моделирования электромеханических и компьютерных систем

9 декабря 2025 г.

Цель и задачи

Цель

Разработать более эффективный планировщик процессов для гетерогенного процессора (Hybrid CPU, big.LITTLE)

Задачи

1. Реализовать алгоритм EEVDF с использованием sched_ext
2. Определить переменные, параметры и ограничения, построить модель
3. Создать тестовое окружение и набор тестовых программ
4. Придумать и реализовать итоговый алгоритм используя sched_ext

sched_ext

Описание

Инфраструктура в ядре Linux для разработки планировщиков на eBPF и их исполнения в виртуальной машине ядра

Особенности

- Предоставляет полноценный API для реализации любого алгоритма планирования
- Динамическое подключение и отключение
- Безопасность: откат на EEVDF при ошибках

Цикл планирования:

1. select_cpu() → выбор потока процессора
2. enqueue() → вставка в глобальную очередь DSQ
3. dispatch() → перемещение в локальную DSQ
4. dequeue() → извлечение из локальной очереди

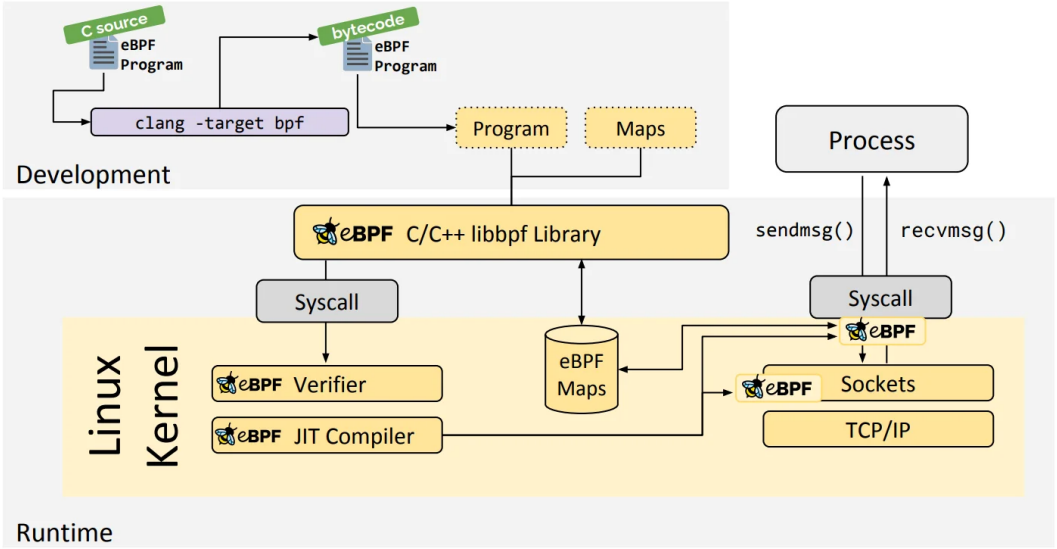

Fig. 1: Схема работы eBPF

Fig. 2: Гетерогенный процессор Intel (целевая платформа)

Введение

- Алгоритм построен на концепции сведения параметров задачи в одну функцию – виртуальное время
- Решение о планировании принимается так: выбирается процесс у которого есть доступный запрос с самым ранним виртуальным дедлайном
- Главный результат: гарантия справедливости алгоритма EEVDF, задержка строго ограничена размером временного кванта q
- Показано, что ограничение является оптимальным для алгоритмов пропорционального распределения и улучшает все предыдущие ограничения

Earliest Eligible Virtual Deadline First : A Flexible and Accurate Mechanism for Proportional Share Resource Allocation

Ion Stoica, Hussein Abdel-Wahab
Old Dominion University 1996

Предположения

- Процессы конкурируют за общий разделяющийся по времени ресурс (CPU).
- Ресурс распределяется во временных квантах не больше чем q.
- В начале каждого кванта времени для использования ресурса выбирается процесс.
- После того как процесс захватил ресурс он может использовать его весь квант времени либо освободить до окончания кванта времени.
- Каждому процессу назначается вес w_i определяющий относительную долю времени которая ему положена.
- Только процессы могут порождать временные кванты.

Модель

$$f_i(t) = \frac{w_i}{\sum_{j \in A(t)} w_j} \quad \text{(client share)}$$

$$lag_i(t) = S_i(t_0, t) - s_i(t_0, t)$$
(ideal minus actual service time)

$$V(t) = \int_{t_0}^{t} \frac{1}{\sum_{j \in A(\tau)} w_j} d\tau$$
(system virtual time)

$$S_i(t_1, t_2) = w_i(V(t_2) - V(t_1))$$
(ideal service time)

$$V(e) = V(t_0) + \frac{s_i(t_0, t)}{w_i}$$
(virtual eligible time)

Модель

$$V(d) = V(e) + \frac{r}{w_i} \quad \texttt{(virtual dedline)}$$

$$V(t^+) = V(t) + \frac{lag_j(t)}{\sum_{i \in A(t^+)} w_i} \quad \texttt{(join)}$$

$$V(t^+) = V(t) + \frac{lag_j(t)}{\sum_{i \in A(t^+)} w_i} - \frac{lag_j(t)}{\sum_{i \in A(t^+)} w_i} \quad \texttt{(weight change)}$$

$$V(t^+) = V(t) - \frac{lag_j(t)}{\sum_{i \in A(t^+)} w_i} \quad \texttt{(leave)}$$

Наивное сравнение

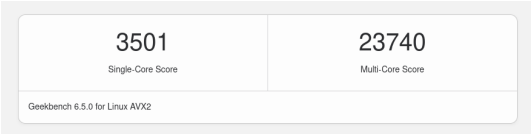

Fig. 3: EEVDF

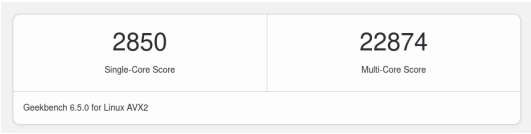

Fig. 4: scx_nest

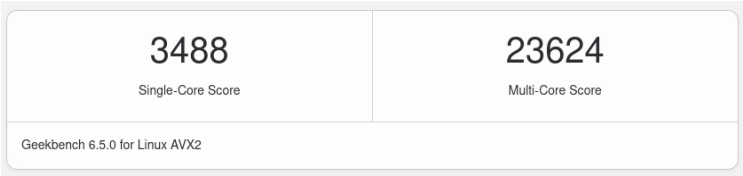

Fig. 5: EEVDF (sched_ext)

Спасибо за внимание, вопросы?

Репозиторий с реализацией

https://github.com/shgpavel/A1349

Дополнительный слайд. Репозиторий scx

Репозиторий с другими sched_ext планировщиками

https://github.com/sched-ext/scx

Дополнительный слайд. Обозначения 1

n – количество процессов

n_a – количество активных процессов

w_i – вес i-го процесса

q – квант времени

m – количество выделенных временных квантов

r – запрошенное процессом время на исполнение

r_{max} – максимально возможное время на исполнение

$r_i^{(k)}$ – продолжительность исполнения k-го запроса i-го процесса

$u_i^{(k)}$ – время которое фактически получает процесс i на k-ом запросе

t – момент в реальном времени

t_a – момент реального времени когда процесс i становится активным

t^- – время прямо перед наступлением события

t^+ – время сразу после наступления события

$A(t)$ – множество всех активных в момент t процессов

$W(t)$ – сумма весов всех активных процессов

Дополнительный слайд. Обозначения 2

$f_i(t)$ – доля процесса i во время t

$S_i(t_0, t_1)$ – время обслуживания которое должен получить процесс i в идеальной системе

$s_i(t_a, t)$ – время обслуживания которое клиент i на самом деле получает

$V(t)$ – виртуальное время системы

e – реальное подходящее время время запроса

d – реальный дедлайн запроса

B – множество активных процессов дедлайн которых в $[e, d]$

C – множество активных процессов дедлайн которых больше d

$ve_i^{(k)}$ – виртуальное подходящее время k-го запроса i-го процесса

$vd_i^{(k)}$ – виртуальный дедлайн k-го запроса i-го процесса

$lag_i(t)$ – задержка обслуживания процесса i в момент времени t, $(S - s)$

d_{neg} – наибольший дедлайн среди всех процессов с отрицательными lag которые активны в момент времени t_1