SKIL Studentenkonferenz Informatik Leipzig
25. September 2012

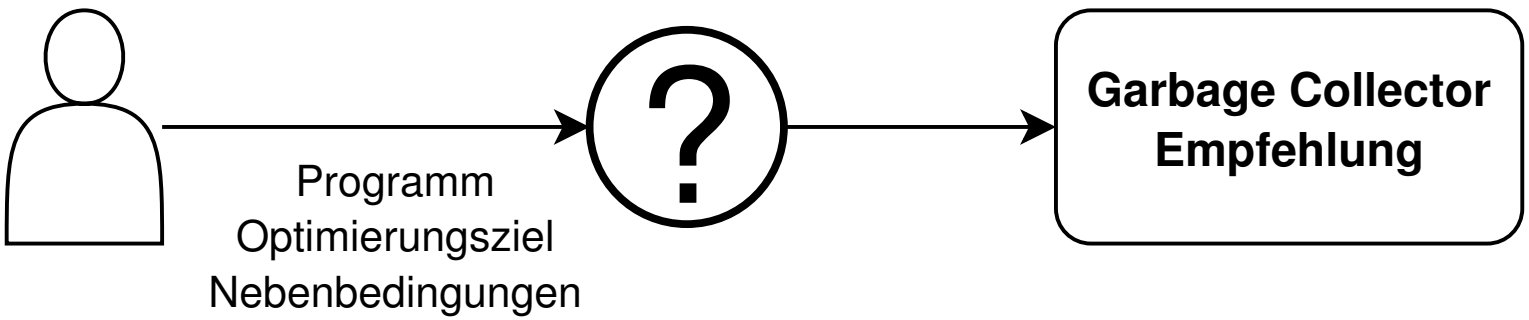
Metriken und optimale Einsatzszenarien für Garbage Collectoren der Java HotSpot Virtual Machine

Michael Schmeißer

michael@skamandros.de

25. September 2012

Ziel


?
Programm
Optimierungsziel
Nebenbedingungen
Garbage Collector
Empfehlung

Motivation

Realbeispiel

- Komponente zur Extraktion von Informationen aus HTML-Dokumenten
- Eingabe: HTML-Dokument
- Ausgabe: Informationen über das Dokument

Fragestellung

Mit welchem Garbage Collector wird der beste Durchsatz erzielt, wenn 512 MiB RAM zur Verfügung stehen?

Was ist Garbage?

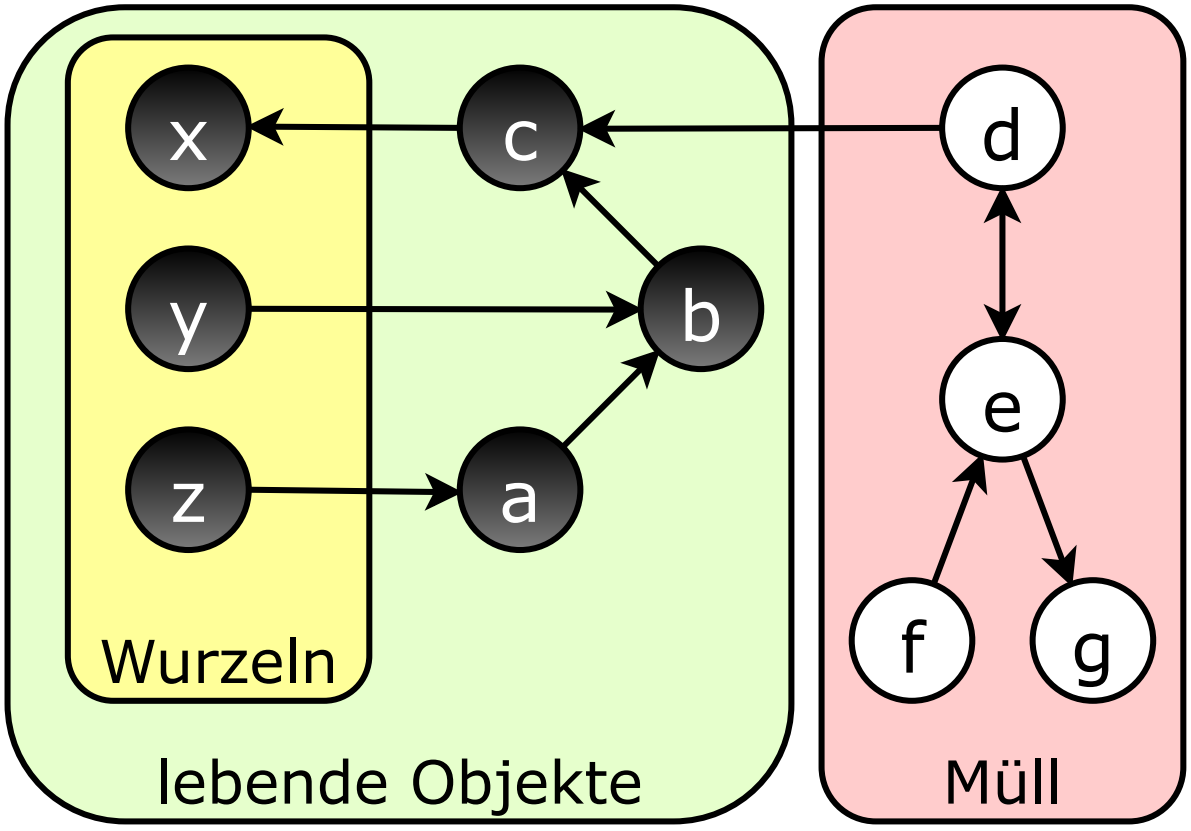


Abbildung: Betrachtung des Heaps als Graph

Garbage Collectoren der Java HotSpot Virtual Machine

- Serial Collector
- Parallel Compacting (PC) Collector
- Concurrent Mark-Sweep (CMS) Collector

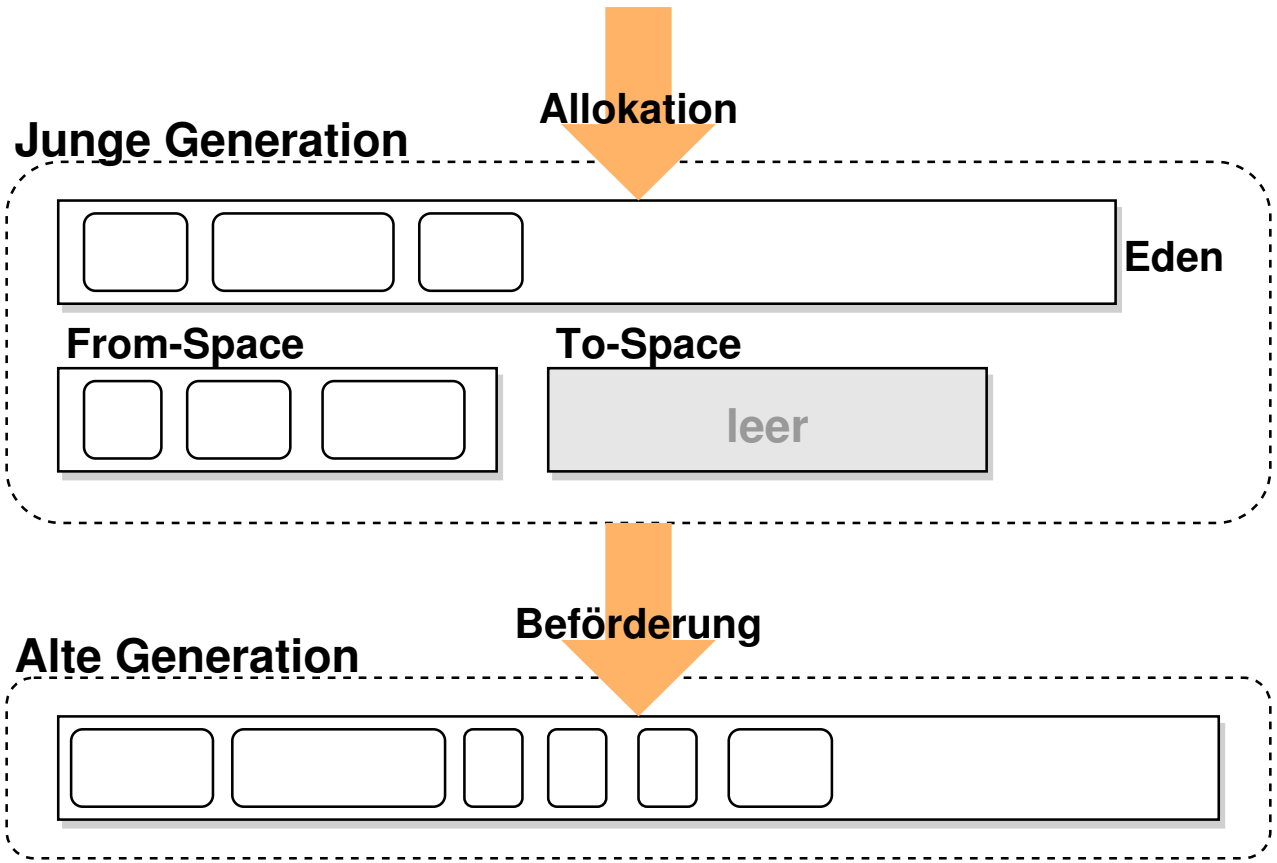


Abbildung: Heap-Layout der betrachteten Garbage Collectoren, vgl. [1, Abb. 1 u. 2, S. 6f.]

Kriterien für die Unterscheidung von Szenarien
Allokationsgeschwindigkeit

- Frage: Wie schnell füllt sich der Heap?
- Allokationsgeschwindigkeit relativ zur verfügbaren Heap-Größe relevant
- Metrik: Anzahl Aufräumzyklen pro Sekunde multipliziert mit Verhältnis der Größe der Allokationsregion zum gesamten Heap

$$v_a = \frac{n}{T} \cdot \frac{s_E}{s} \tag{1}$$

- Bei konstantem Verhältnis $\frac{s_E}{s}$:

$$v'_a = \frac{n}{T} \tag{2}$$

Kriterien für die Unterscheidung von Szenarien
Lebenszeit der Objekte

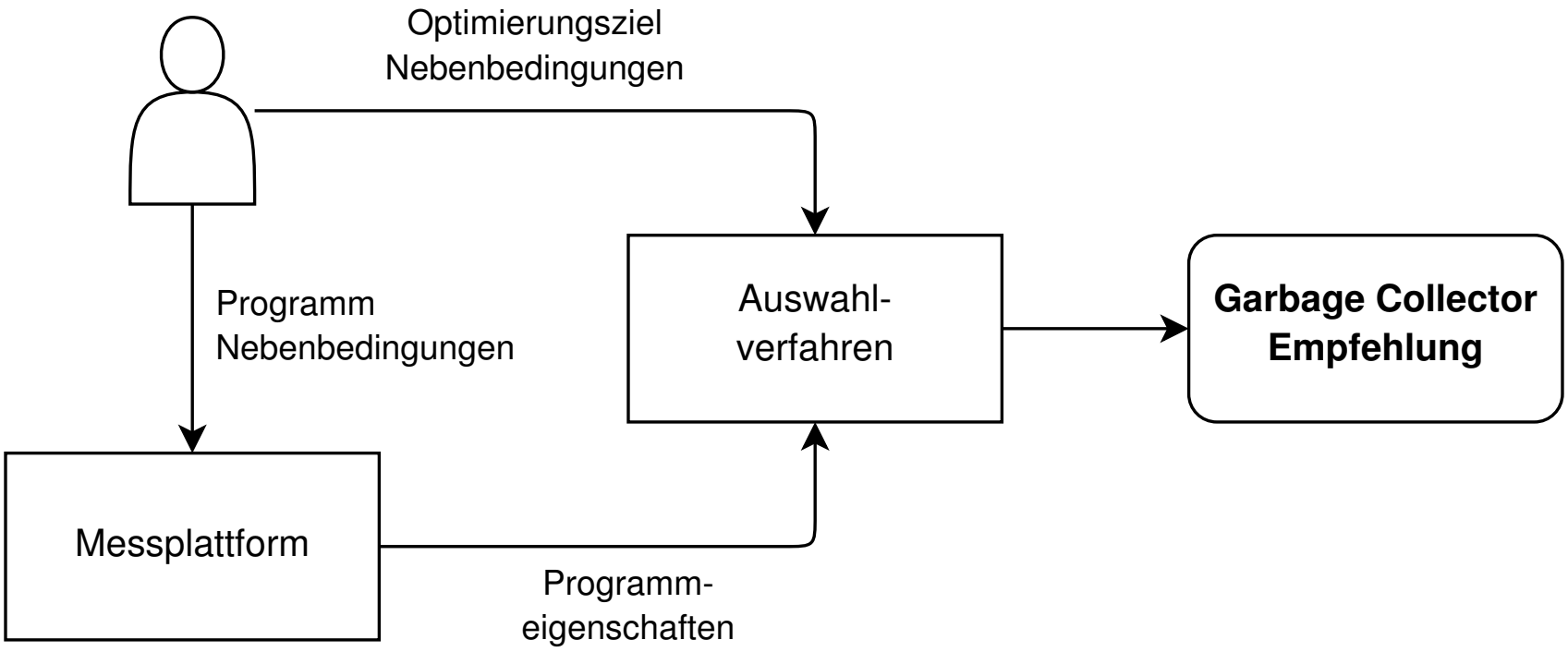
Schwache generationale Hypothese [2]

Die meisten Objekte sterben jung.

- Garbage Collectoren für schwache generationale Hypothese optimiert
- Frage: Wie lange leben die Objekte?
- Problem: Große Aufräumzyklen führen zu schlechter Performance bei pausierenden Kollektoren
- Metrik: relative Häufigkeit großer Aufräumzyklen

$$\Gamma = \frac{H(G)}{H(G) + H(K)} \tag{3}$$

Gesamtprozess



Ermittlung optimaler Einsatzszenarien
Methodik

- Durchführung mehrerer Experimente mit verschiedenen Programmen
- Beeinflussung der Allokationsgeschwindigkeit und Lebenszeit der Daten
- Gruppierung der Ergebnisse in verschiedene Intervalle für v'_a und Γ
- Gesamtergebnis: schlechtestes relatives Ergebnis

$$\text{Symbol} = \begin{cases} ++ & \text{stets bestes Ergebnis} \\ + & \text{nie mehr als 10\% schlechter} \\ \circ & \text{nie mehr als 20\% schlechter} \\ - & \text{nie mehr als 30\% schlechter} \\ - - & \text{sonst} \end{cases} \tag{4}$$

Einordnung des Realbeispiels

Heap-Größe [MiB]	$v'_a \left[\frac{\text{Aufräumzyklen}}{\text{Sekunde}}\right]$	Γ [%]
512	4,6	0,3

Tabelle: Eigenschaften des Realbeispiels

Serial Collector	PC Collector	CMS Collector
+	++	○

Tabelle: Gesamtergebnisse der Garbage Collectoren bezüglich Gesamtausführungszeit bei $2,5 < v'_a \leq 10,0 \wedge \Gamma \leq 0,5\%$

Auswahl eines Garbage Collectors für das Realbeispiel

Serial Collector	PC Collector	CMS Collector
+	++	○

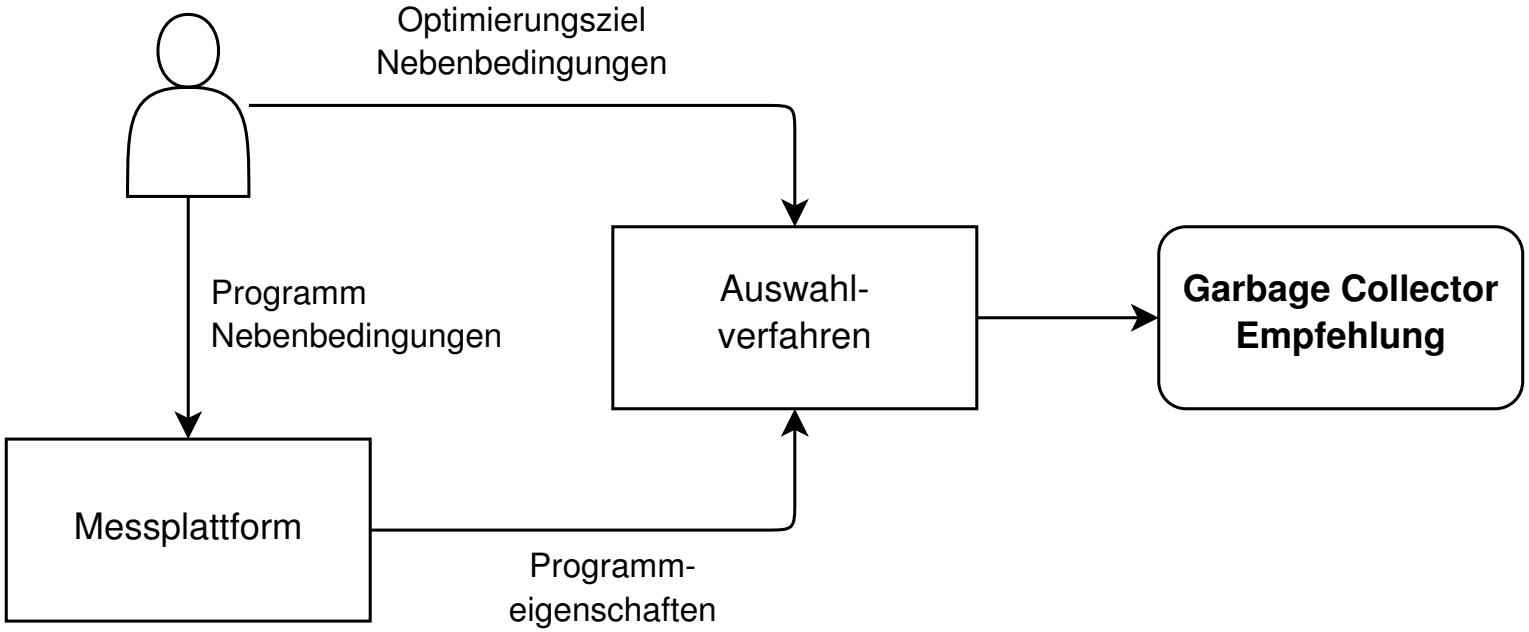
Tabelle: Gesamtergebnisse der Garbage Collectoren bezüglich Gesamtausführungszeit bei $2,5 < v'_a \leq 10,0 \wedge \Gamma \leq 0,5\%$

Serial Collector		PC Collector		CMS Collector	
$\overline{T}$	σ_T	$\overline{T}$	σ_T	$\overline{T}$	σ_T
4 773	32	4 681	12	4 919	32

Tabelle: Gesamtausführungszeit T in Millisekunden ermittelt bei 512 MiB Heap-Größe in 240 Versuchen mit Entfernung von jeweils 34 Minima und Maxima

Zusammenfassung

- Ziel: Auswahl eines geeigneten Garbage Collectors mit möglichst wenig Aufwand
- Vorgehen: Durchführung verschiedener Experimente – Gruppierung der Ergebnisse zur Verallgemeinerung
- Ergebnis: Optimale Einsatzszenarien für die Garbage Collectoren der HotSpot JVM
- Anwendung: Auswahl eines GCs durch Bestimmung von v'_a und Γ



Literatur

SUN MICROSYSTEMS:
Memory Management in the Java HotSpot™ Virtual Machine.
http://java.sun.com/j2se/reference/whitepapers/memorymanagement_whitepaper.pdf, April 2006

UNGAR, David M.:
Generation Scavenging: A Non-Disruptive High Performance Storage Reclamation Algorithm.
In: *ACM/SIGSOFT/SIGPLAN Software Engineering Symposium on Practical Software Development Environments*.
Pittsburgh, PA : ACM Press, April 1984 (ACM SIGPLAN Notices 19(5)), S. 157–167