Universität Dortmund
PG 431“Metaheuristiken“

Fahrstuhlproblem

Problemstellung und Lösungsansätze

Vedran Divkovic

Dortmund, April 2003

Inhaltsverzeichnis

1 Geschichte

Es scheint unglaublich, aber die ersten Aufzüge gab es schon 400 vor Christus. Sie wurden damals in griechischen Theatern eingesetzt, um einen als Gott verkleideten Schauspieler auf die Bühne herabzulassen.
Sogar Kaiser Nero benutzte in Rom einen Aufzug zwischen zwei Stockwerken. Erfinder dieser ersten Aufzüge sind Aristoteles und Archimedes.
Die größte Entwicklung haben Aufzüge in den letzten 200 Jahren erlebt. Moderne Aufzüge von heute haben nicht nur ein Paar Buttons sondern hoch entwickelte Keypads.
Früher war man froh wenigstens einen einzigen Aufzug im Haus zu haben, heute übersteigt manchmal die Anzahl der Aufzüge selbst die Anzahl der Etagen.

Hier die wichtigsten Aufzugserfindungen:

1834	Erfindung des Drahtseilerei
1853	Fangvorrichtung durch Elisha Otis
1876	Geschwindigkeitsbegrenzer
1877	Anwendung der Treibscheibe für Vertikaltransport durch Carl Friedrich Koepe
1878	Indirekter Hydraulik-Aufzug
1880	elektrischer Antrieb durch Werner von Siemens
1888	Einführung der Seilkonstruktionen ‚Seale', ‚Warrington' und ‚Filler'
1889	elektrischer Antrieb für Trommelwinden über Schneckengetriebe
1892	Ward-Leonard-Antrieb
1894	Druckknopfsteuerung
1903	Getriebeloser Treibscheibenaufzug
ab 1948	Automatische Programmierung der Aufzüge

2 Einleitung

Jeden morgen passiert es: ich verschlafe, fahre durch die Stadt und komme dann in mein Bürogebäude, warte ungeduldig auf den Aufzug, doch dieser will einfach nicht kommen - er kommt eigentlich nie wenn man ihn braucht! Jedes mal frage ich mich, wer wohl diese Steuerung entworfen hat…
Doch man muss auch den Aufzug ein wenig „verstehen“, wenn er z.B. in der 4. Etage steht. Eine Person will von der dritten nach unten befördert werden, eine andere nach oben und eine dritte von der allerletzten Etage ganz nach unten.
Wie weiß ein Aufzug wo er zuerst hinfahren und welche Person er zuerst befördern soll?
Versetzen wir uns in die Lage des Aufzugs, der gerade im achten Stock steht. Soeben hat A im siebten Stock einen Transport nach unten angefordert, B im dritten Stock nach oben und C im zwölften nach unten. Wir haben nur sehr wenig Zeit zu überlegen („Echt-Zeit“), damit der Geduldsfaden der Wartenden nicht reißt. Doch als wir gerade in der achten Etage losgefahren sind, drückt D auf der achten Etage auf den Knopf. Was nun? Zurückfahren? Was, wenn nach Aufsammeln von D in der achten Etage auch noch E, F, G mitfahren wollen? Aber wann holen wir dann C ab? Also sollen wir doch lieber D warten lassen, bis alle bisherigen Aufträge abgearbeitet sind und D sauer wird, weil wir schon zum wiederholten Mal an seine Etage ungnädig vorbeirauschen?

Diese Problemstellung werde ich in meiner Arbeit darstellen und versuchen die verschiedenen Lösungsmöglichkeiten dafür zu beschreiben.

Kehren wir zum Aufzugsproblem zurück. Um Lösungen sinnvoll bewerten zu können, benötigt man eine Zielfunktion. Bei Personenaufzügen sind zufriedene Benutzer oberste Priorität. Doch hier stellt sich wieder die Frage, ob die individuelle Zufriedenheit überhaupt erreicht werden kann. Denn es wird immer wieder jemanden geben, der unzufrieden sein wird. Wäre vielleicht das *„First-Come-First-Serve-Prinzip“* eine Lösung? – von der Logik her ja, wenn es nicht so viele Passagiere zu befördern gibt, doch bei stärkerer Auslastung liefert dieses System sehr ineffiziente Fahrpläne und so werden auch die individuellen Wartezeiten untragbar.

3 Elevator Supervisory Group Control Problem

3.1 System - Architektur

Ein Aufzugsystem besteht aus einem *Elevator Group Controller* und einem oder mehreren *Elevator Controllers*.

- *Elevator Group Controller* (EGC) bestimmt zu welcher Etage einzelne Aufzüge fahren sollen und welche Passagiere zuerst bedient werden
- *Elevator Controllers* (EC) sind für Funktionen innerhalb eines Aufzugs zuständig wie z.B. Türkontrolle oder Messung des Ladezustands

3.2 Elevator Group Controller (EGC)

Der EGC kann durch folgende Punkte definiert werden:

- Die Ankunft der Passagiere folgt dem Prinzip der *Poisson*-Verteilung: Die *Poisson*-Verteilung beschreibt die Wahrscheinlichkeit, dass in einem bestimmten Zeitabschnitt eine gegebene Anzahl von Ereignissen stattfindet
- Der *Group Controller* hat die Aufgabe die Aufzugsaufrufe an die einzelnen Aufzüge weiterzuleiten
- Die *Hall call time* ist die Zeitspanne bis sich ein Aufzug zu der Etage bewegt
- Die *Passenger waiting time* ist die Zeitspanne von der Ankunft des Passagiers bis zum Einsteigen in den Aufzug
- Die *Passenger ride time* ist die Zeitspanne vom Einsteigen bis zum Aussteigen
- Die *Service time* ist definiert als Summe aus Fahr- und Wartezeit
- Die *Round trip time* ist die Zeit die ein Aufzug für eine ganze Fahrrunde benötigt

3.3 ESGC Problem

In diesem Abschnitt beschreibe ich das *Elevator Supervisory Group Control Problem*. Wie bereits gesagt bestimmt ein *Elevator Group Controller* zu welchen Etagen die Aufzüge fahren sollen.

Die optimale Kontrollstrategie für ein Aufzugsystem ist die Wartezeiten der Passagiere auf Aufzüge zu minimieren und die Kapazitäten der Aufzüge zu maximieren.
Das erste Problem dabei ist, das es für verschiedene Gebäude und zu verschiedenen Zeitpunkten auch verschiedene Konfigurationen der Aufzüge geben muss. Dieses Problem wurde von einem der größten Aufzugshersteller gelöst. Dieser Hersteller hat einen auf den neuralen Netzen basierten Kontroller (*neural network based controller*) entwickelt mit dessen Hilfe man für verschiedene Auslastungen des Verkehrs die besten Kontroll-Strategien bekommt.

Das eine Problem bleibt aber immer noch ungelöst und zwar „*Wie können Aufzüge den Passagieren in „Echt-Zeit" zugeteilt werden während gleichzeitig:*

- *Die gesamte Service-Qualität,*
- *Der Verkehrdurchsatz,*
- *Der Energieverbrauch und*
- *Der Bedarf am individuellen Service für einzelne Passagiere(gruppen)*

…optimiert werden?"

Und genau dieses Problem ist ein Teil des *Elevator Supervisory Group Controls* (ESGC).

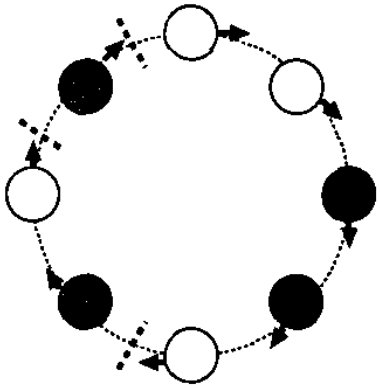
Das ESGC Problem hat sich durch die Zeit verändert. Im Kontrast zu den traditionellen Aufzügen wo die Passagiere vor dem Eingang ein Button (🡅 oder 🡇) gedrückt haben und erst dann im Aufzug die gewünschte Etage, ist es heute so dass wir bei modernen Aufzugssystemen vor dem Eingang nicht einfache Buttons haben, sondern funktionsreiche *Keypads*, wo wir z.B. gleich den Bestimmungsort angeben können.
Dies liefert mehr exakte Informationen für den *Group Controller*, schränkt aber gleichzeitig seine Entscheidungsfreiheit ein.
Derzeitige Forschungen versuchen auf verschiedenen Wegen das beste Optimierungssystem für das ESGC Problem zu finden. Heute existieren Berichte über sehr gute Ergebnisse. Allerdings sind alle diese Methoden miteinander nicht kompatibel und nicht reproduzierbar. Die Ursachen dafür liegen in der Tatsache, das sich die einzelnen Aufzugssysteme an denen man forscht voneinander

unterscheiden, und das sie viele unterschiedliche Parameter haben (beispielsweise die Anzahl der Aufzüge, Anzahl der Etagen, Verkehrsauslastungen.)
Hier wird ein vereinfachtes, kompatibles und leicht reproduzierbares Aufzugs-Modell vorgestellt, das auch Forschern in ganz unterschiedlichen Forschungsgebieten helfen soll. Es ist also nicht nur auf das ESGC Problem beschränkt. Wir sprechen hier vom S-Ring-Modell.

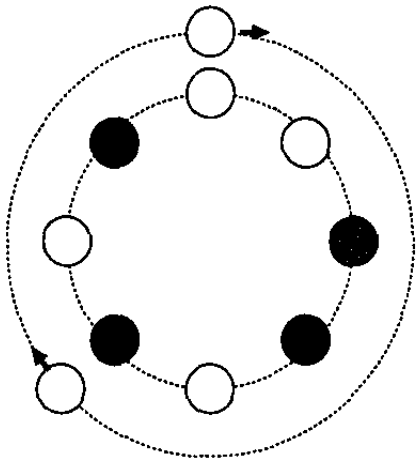
4 S-Ring

4.1 Einführung

Wir benutzen den S-Ring als ein vereinfachtes Modell eines realen Aufzugs und versuchen mit dessen Hilfe das ESGC Problem zu lösen.
Unser S-Ring-Modell basiert auf dem Kac-Ring, das vom Marc Kac erfunden wurde.



Kac-Ring



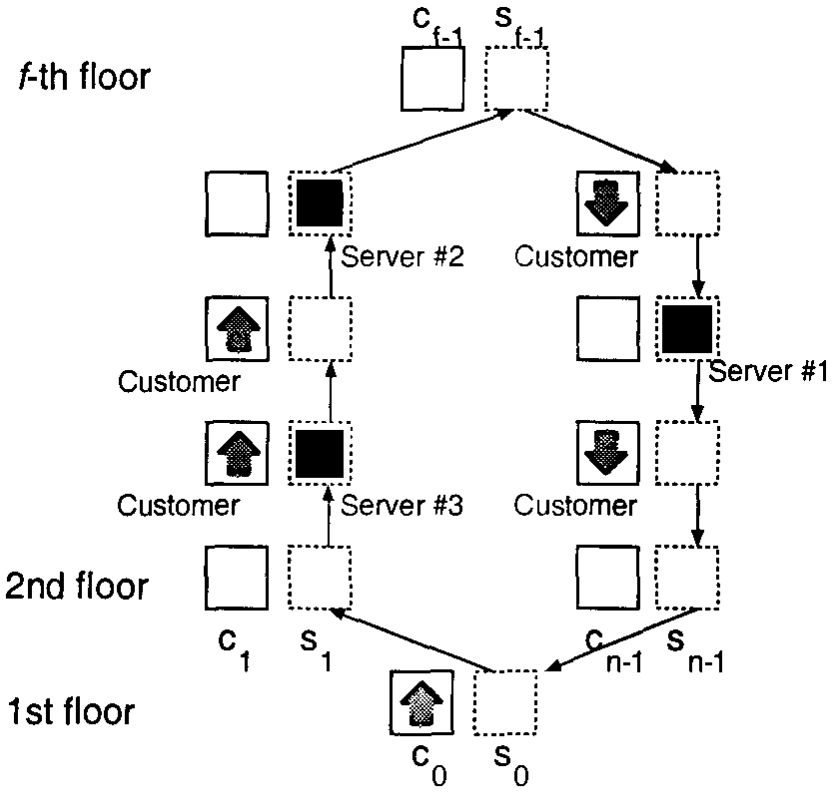
S-Ring

Der S-Ring unterscheidet sich vom Kac-Ring dadurch dass zusätzlich noch zwei weitere Eigenschaften zugefügt wurden und zwar: die stochastische Zustandsänderung und die Kontrolle.
Der S-Ring liefert die exakten Ergebnisse nur für kleine Probleme. Leicht reproduzierbar bedeutet dass man den S-Ring auch für andere Probleme anwenden kann. Es ist keinesfalls nur auf unser Aufzugsproblem beschränkt.

4.2. S-Ring-Parameter

- Aufzug (s_i)
- Passagier (c_i)
- Ankunftsrate (p)
- Anzahl der Etagen (n)
- Anzahl der Aufzüge (m)
- Policy (¶)
- Fitnessfunktion (Q)

4.3. S-Ring als Aufzugssystem



Hier haben wir zwei Argumente: den Passagier (c_i) und den Aufzug (s_i).

c_i = 1 bedeutet das es einen oder mehrere wartende Passagiere auf der i-ten Etage gibt. Logischerweise gibt es für den Wert 0 keine Passagiere.

s_i =1 bedeutet das ein Aufzug auf der i-ten Etage angekommen ist. Wie auch oben beschrieben gibt es für den Wert 0 keinen Aufzug.

Mit diesen 2 Parametern kann man den Zustand des Systems zum Zeitpunkt *t* definieren.

$$[s_0(t), c_0(t),\ldots s_{n-1}(t), c_{n-1}(t)] \equiv x(t) \in X = \{0,1\}^{2n}$$

€ = Element aus…

Wir haben also *n* Etagen und jede ist mit einem 2 bit Zustand (c_i,,s_i) identifizierbar. Bei unserem S-Ring Modell scannen wir Etagen von *n-1* bis 0 und dann wieder von *n-1*. In jedem Schritt wird der Trippel (c_i, s_i*i*, s_{i+1}) aktualisiert.

4.4 Zustandstabelle eines S-Rings

ε(t)	Prob	¶ (x)	ε(t+1)	∆r
000	1-p		000	0
	p		100	-1
100	1		100	0
010	1-p		001	0
	p	0	101	-1
		1	010	0
110	1	0	101	0
		1	010	+1
001	1-p		001	0
	p		101	-1
101	1		101	0
011	1		011	0
111	1		011	+1

ε(t): der aktuelle Zustand

Prob: Wahrscheinlichkeit, dass es auf dieser Etage einen wartenden Passagier gibt

¶ (x): *policy* ¶

ε(t+1): Folgezustand

∆r: Differenz der Anzahl wartender Passagiere

Ein Gebäude hat *f* Etagen, also ist die Anzahl der Listen von wartenden Passagieren **n=2f-2**.

Wenn die Liste einen Passagier hat und es einen Aufzug auf der gleichen Etage gibt, dann muss der Aufzug den Passagier entweder mitnehmen (TAKE) oder zurücklassen (PASS).

Wann ist das der Fall? Wenn wir einen wartenden Passagier haben (1XX) und wenn wir einen Aufzug auf derselben Etage haben (11X), aber keinen Aufzug auf der nächsten Etage (111).
Die letzte Bedingung folgt aus der Eigenschaft des S-Rings, die besagt dass zwei Aufzüge einander nicht passieren dürfen.
Für die Listen in denen keine Passagiere oder Aufzüge vorhanden sind, braucht man gar nicht die *policy* ¶ anzugeben.

4.5 Unterschiede zwischen einem Aufzug und dem S-Ring

Allgemein lässt sich sagen dass Aufzüge nach dem Prinzip eines Kreislaufs funktionieren. Im Detail bedeutet das dass sie zuerst hinauf gehen, dann hinunter und wieder hinauf, je nach Bedarf wo der Benutzer einsteigen bzw. aussteigen will. Das S-Ring-Modell basiert auf dieser ganz einfachen Betrachtungsweise, aber um es zu präzisieren müssen wir die Unterschiede zwischen den realen Aufzügen und dem Modell explizit definieren:

- Modell-Aufzüge können Passagiere aufnehmen, sie aber nicht entladen
- Modell-Aufzüge haben unbeschränkte Kapazität, was bedeutet dass sie unendlich viele Passagiere aufnehmen können. In der Realität existieren jedoch Beschränkungen wie z.B. das nur eine bestimmte Anzahl von Passagieren in den Aufzug einsteigen darf
- Es werden keine Statistiken über die geladenen Passagiere geführt
- Modell-Aufzüge haben identische Passagierankunftsraten für alle Etagen und Fahrrichtungen, und sind durch *p* gekennzeichnet
- Alle Etagen sind gleich voneinander entfernt
- Die einzelnen Etagen sind durch ein bit c_i und die einzelnen Aufzüge durch ein bit s_i identifizierbar
- Aufzüge haben gleiche Fahr- und Stoppzeiten
- Zeit ist diskret, Ereignisse finden in Zeitintervallen $t \in \{0,1,\ldots\}$ statt
- Das Modell hat noch zusätzlich die *policy* ¶ : $x \rightarrow \{0,1\}$

5 S-Ring und mögliche Lösungen

Es stehen uns mehrere Verfahren zum Erreichen einer Lösung zur Verfügung und zwar:

- Die Lösung durch dynamische Programmierung
- Die Lösung durch numerische Methoden:
    - *Q-learning*
    - *Kiefer- Wolfowitz stochastische Approximation*
    - *Evolutionary Strategies (*ES*)* (diese wird später näher erläutert)

5.1 Vergleiche zwischen KW, Q-l und ES

Um die 3 Methoden miteinander zu vergleichen muss man mit unterschiedlichen Werten für *p* arbeiten (*p* = Ankunftsrate)

- ES ist mindestens so gut wie KW
- Q-l hat die gleiche oder sogar bessere Ergebnisse als ES, aber auf Kosten anderer Funktionen

Aus diesen Vergleichen und praktischen Ergebnissen nehmen wir die ES als die optimale Lösung für unser Kontrollproblem.

5.2 Optimierung eines S-Rings durch ES

Der S-Ring wird benutzt um das optimale Kontrollproblem (*optimal controll problem*) zu definieren.

5.2.1 Policy ¶

Die *Policy* ¶ ist die Wahrscheinlichkeit dass ein Passagier bei Vorhandensein eines Aufzugs genommen wird.
Für die gegebenen Parameter *n*, *m* und *p* ist die Systementwicklung nur von der *policy* ¶ abhängig. Das kann geschrieben werden als

$$\P^* = \arg\min Q(\P)$$

Das Hauptproblem dabei ist es für die vorgegebenen Werte von *n*, *m* und *p* die optimale *policy* ¶ zu berechnen. ¶ steigt exponentiell mit der Anzahl der Etagen *n*.
Im Prinzip berechnet man $\P^*$ indem man zunächst alle möglichen ¶ berechnet und dann eins mit dem größtem *Q* nimmt.

5.2.2 Beispiele für policy ¶

Die einfachste *policy* ist die „*greedy*"- policy:

$$\P^g = 1$$

Das bedeutet dass jeder Passagier bei Vorhandensein eines Aufzugs sofort bedient wird. Diese *policy* ist aber nicht optimal, außer im Falle eines großen Andrangs (*p*>0.5). Daher muss eine gute policy einige Passagiere gelegentlich umgehen.
Ein Beispiel dafür ist die „*balance*"- policy:

$$\P^b = \begin{cases} 0 \\ 1 \end{cases} \text{ wenn } s_{n-1} = 1 \text{ und 1 sonst.}$$

Die Idee bei der „*balance*"- policy ist es die Aufzüge besser *zu verstreuen*, während bei der „*greedy*"- policy die Aufzüge näher zueinander stehen.

6 Fazit

In zukünftigen Arbeiten wird man versuchen eine optimale ES-Variante zu finden, die eine größere Anzahl an Parametern abdeckt.
Am Ende findet man den S-Ring bei der Auswahl und Entwicklung der optimalen Methoden für dynamisch kontrollierte Systeme als sehr hilfreich. Man hofft das der S-Ring auch für andere Forschungen in diesem Bereich nützlich werden könnte.

7 Quellenverzeichnis

Sandor Markon, Yoshikazu Nishikawa. *On the Analysis and Optimization of Dynamical Cellular Automata with Application to Elevator Control.* 2002

Sandor Markon, Dirk V. Arnold, Thomas Bäck, Thomas Beielstein, Hans-Georg Beyer*. Thresholding – a Selection Operator for Noisy ES.*

Thomas Beielstein, Claus Peter Ewald, Sandor Markon. *GECCO2003*

M. L. Siikonen*. Planing and Control Models for Elevators in High-Rise Buildings.* 1997

Thomas Beielstein. *Threshold Selection, Hypothesis Tests, and DOE Methods*. 2002

G. Barney. *Elevator Traffic Analysis, Design and Control.* 1986