3. Objekt-Experimente; Prinzipien und statistische Grundlagen

Motivation und Anschluß

Zurück zum Ende von Kap. 1: Es geht uns bei der Leistungsbewertung von RS allgemein darum, den Zusammenhang zwischen den Einflußgrößen "Maschine" und "Last" einerseits und den resultierenden Werten von "Leistungsmaßen" andererseits aufzudecken. Idealerweise, hatten wir entdeckt, wäre dieser Zusammenhang durch "Leistungsfunktionen" L der Art (1.18) vollständig beschrieben. Es erschien uns allerdings zweifelhaft, ob geschlossene mathematische Formen für L (in allgemein gelagerten Fällen) zu finden sein würden. Wir hatten uns daher nach einem "zweitbesten" Weg umgetan und einen experimentellen Ansatz als hoffnungsvoll empfunden: Dieser zielte nicht auf eine geschlossene Form für L ab sondern (bescheidener) auf Möglichkeiten, die Funktion L "punktweise abzutasten", d.h. für jedes konkrete (Maschine,Last)-Paar den zugehörigen Leistungswert zu ermitteln. "Bescheidener" war dieser Ansatz einzuschätzen, da komplexere Fragestellungen (bei Entwurf, Optimierung, ...) von vornherein auf eine gewisse Anzahl solcher Abtastungen ("Versuche", "Experimente") verwiesen waren. "Hoffnungsvoll" erschien uns dieser Ansatz, weil uns prinzipielle Realisierungsmöglichkeiten unmittelbar einfielen: Zum einen die Beobachtung (Messung) des zu bewertenden RS selbst, zum anderen die Beobachtung simulativer (und eventuell anderer) Modelle des RS. Wir verlegen uns in diesem Kapitel völlig auf das Experimentieren mit dem zu bewertenden Rechensystem, also auf die in Kap. 1 eingeführten Objekt-Experimente. Eine der Grundlagen für Objekt-Experimente haben wir uns inzwischen in Kap. 2 erarbeitet in Form diverser Ansätze und Instrumente zur Messung von RS.

Lernziele

Wir werden im Verlauf dieses Kapitels, und dies bezeichnet auch direkt unsere Lernziele,

- die Prinzipien der Durchführung von Objekt-Experimenten und der Interpretation dabei gewonnener Meßwerte kennenlernen;
- verstehen, daß sowohl bei der Charakterisierung von Lasten als auch bei der Bewertung von Experimenten statistische Methoden eine zentrale Rolle spielen;
- einige statistische Grundlagen wiederholen und zugehörige Fertigkeiten üben;
- eine Übersicht über Möglichkeiten der Last-Darstellung zum Zwecke der Durchführung von Experimenten gewinnen.

Voraussetzungen

Statistik wird hier und in folgenden Kapiteln eine Rolle spielen. Wir können im Rahmen dieses Kurses sicher nicht die notwendigen statistischen Grundlagen einführen. Vielmehr vertraue ich darauf, daß Sie diese Grundlagenkenntnisse besitzen oder sich (auch zur evtl. Auffrischung Ihrer Kenntnisse) mit der einschlägigen Literatur auseinandersetzen. Ich empfehle als guten Einstieg in Stochastik und Statistik Dörf78, Kap. 10; als gerne verwendetes Lehrbuch der Statistik MoGr63; als Nachschlagewerk für häufigen Gebrauch HaEK82. Ein Besuch einer wissenschaftlichen Bibliothek (immer wieder empfehlenswert!) wird Ihnen aber sehr schnell zeigen, daß es eine Vielzahl von einführenden und vertiefenden Werken im Bereich der Stochastik und Statistik gibt. Wählen Sie ggf. "nach Geschmack".

3.1 Prinzipielle Überlegungen

Fragestellungen

Wir sind dabei, die Leistungsbewertung von RS per Objekt-Experiment zu durchdenken. Wir erinnern uns (vgl. Kap. 1) an mögliche Fragestellungen:

(i) **Feststellung:** Wie ist derzeit ("heute") die Leistung des zu beurteilenden RS?

(ii) **Exploration:** Wie wäre die Leistung des RS, falls sich folgendes änderte: "Etwas" an der Maschine (eine Hardware-Änderung, System-Software-Änderung, Anwendungs-Software-Änderung, ...) oder an der Last (steigende oder fallende Benutzung bestimmter Programme, neue Anwendungen, ...)?

(iii) **Verbesserung:** Was ist zu tun (Änderungen der Maschine und / oder der Last), um eine bestimmte Leistung zu erzielen, oder die bestmögliche Leistung zu erzielen, oder ...?

Vorgang Objekt-Experiment

Die Beantwortung solcher Fragen per Objekt-Experiment erfordert Beobachtungen eines RS während des Betriebs (jeweils während eines gewissen Meßintervalls und unter Verwendung einer geeigneten Meßinstrumentierung; vgl. Abschn. 2.1); bei Fragestellung (i) wird nur eine solche Beobachtung involviert sein (am System "wie es ist"), bei Fragestellung (ii) i.allg. eine zusätzliche Beobachtung (am gemäß Fragestellung konkret "geänderten" System), bei Fragestellung (iii) höchstwahrscheinlich einige zusätzliche Beobachtungen (jede auf eine spezielle Änderungs-Idee bezogen, und am dementsprechend konkret geänderten System). Halten wir als ein Merkmal von Objekt-Experimenten die offensichtliche Tatsache fest, daß sie mit einem nennenswerten (oft großen) Aufwand verbunden sind. Dieser fällt einerseits auf seiten der Meßinstrumentierung an, andererseits aber in Form der Vorbereitung einer Beobachtung, die ja, zumindest bei Fragestellungen (ii,iii), konkret geänderte, betriebsfähige RS bereitzustellen hat (man denke an die "versuchsweise" Erstellung und Installierung neuer Betriebssystem-Routinen, an den "versuchsweisen" Austausch von Prozessoren, Peripheriegeräten, ...).

Aufwand

Leistungs-kriterien: Beispiele

Gehen wir weiter, zur Beobachtung selbst. Nehmen wir als Beispiel an, unser Interesse gälte einem größeren Mehrbenutzer-RS, und da, genauer gesagt, zwei Leistungskriterien:

Verweilzeiten

(a) den Verweilzeiten (auch "turnaround times") von batch jobs, einem typischen Effektivitäts-Kriterium an der Benutzer / System-Schnittstelle;

Auslastung

(b) den aus Abschn. 2.1 wohlbekannten busy periods einer CPU, aus denen wir in (2.1.7,2.1.15) die Auslastung (relativer busy-Zeitanteil) dieser Maschinenkomponente abzuleiten versuchten, also ein typisches Effizienz-Kriterium.

Reallast

Unser RS ist vereinbarungsgemäß betriebsfähig. Beobachten wir es also während seines normalen Betriebs. Man beachte: Diese Beobachtung erfolgt insbesondere ohne Beeinflussung der Benutzer, d.h. ohne Beeinflussung der von ihnen initiierten System-Last; es ist eine Beobachtung unter **Reallast / live load**. In Verfolgung etwa des Kriteriums (a) werden wir versuchen, als Basis-Information die Verweilzeiten einzelner batch jobs zu messen, so daß zu Ende der Beobachtung eine Menge von job-Verweilzeiten $v_i;i=1,2,\ldots,n$ zur Verfügung steht (Indizierung z.B. in der Reihenfolge der "Fertigstellung" der jobs; n entsprechend der im Meßintervall beobachteten jobs).

Testfrage 3.1.1: Welche unmittelbaren Beobachtungen müssen Sie vornehmen? Welchen Meß-Ansatz verwenden Sie? Welchen Monitor-Typ können Sie einsetzen? Können Sie alle Beobachtungen zur Ermittlung der geforderten v_i-Folge nutzen? Ziehen Sie Kap. 2 zu Rate.

Kritik

Die gemessene Information ($v_1, v_2, \ldots, v_n$) kann uns in verschiedener Hinsicht nicht so recht befriedigen; naheliegende Kritikpunkte sind

Unhandlichkeit

- die **Unhandlichkeit** der Verweilzeitfolge, die in ihrer Gesamtheit für Leistungsbeurteilungen kaum brauchbar ist; wir hatten in Kap. 2 schon angesprochen, daß uns kompaktere Zusammenfassungen dieser Information willkommener wären (z.B. so etwas wie "mittlere Verweilzeit");

Unsicherheit

- die **Unsicherheit** bzgl. einer korrekten Interpretation der Meßergebnisse, da
  - die Verweilzeit eines jobs sicher von seinem Bedarf an Betriebsmitteln abhängt, so daß ein direkter Vergleich einer Verweilzeit v_i mit irgendeiner anderen Verweilzeit v_j ohne weitere Angaben über jobs i,j ziemlich sinnlos ist (wer "viel verlangt", wird auch "länger brauchen");
  - die Verweilzeit eines jobs sicher vom Bedarf an Betriebsmitteln anderer, gleichzeitig "anwesender" jobs abhängt, darüber hinaus auch von den Anforderungen anderer Last-Typen (gleichzeitige interaktive Belastung, …), so daß eine Verweilzeit v_i ohne Betrachtung der sonstigen Belastung ebenfalls wenig aussagt (wenn "viel los" ist, werden auch alle "länger brauchen").

Beeinflussung durch Lastumgebung

Diskutieren wir die verschiedenen Kritikpunkte. Zunächst zur Beeinflussung einer konkreten Verweilzeit v_i durch die "sonstige" Last. Wir wissen aus eigener Erfahrung, bzw. können durch Nachfragen beim Rechenzentrums-Personal oder aus einer Durchsicht der seitens eines Rechenzentrums üblicherweise geführten Betriebs-Statistiken leicht erfahren, daß die Belastung eines Mehrbenutzer-RS stark schwankt; zum einen aufgrund der Lebens- und Arbeits-Gewohnheiten der Benutzer, zum anderen aufgrund bewußt vorgenommener Betriebs-Regelungen. So ist etwa die Benutzung des Systems morgens um 5 Uhr, am späten Vormittag, abends nach 22 Uhr, werktags, am Wochenende, … ziemlich unterschiedlich - aus verständlichen Gründen. Auch gibt es üblicherweise Beschränkungen der Systembenutzung wie maximal zugelassene CPU-Zeiten, Arbeitsspeicher-Grössen, … für bestimmte Tages-Intervalle oder Wochentage, wodurch die Unterschiedlichkeit der tatsächlichen Benutzung des Systems in verschiedenen Zeitintervallen noch zusätzlich gesteuert wird. Glücklicherweise entdeckt man wiederkehrende, sich weitgehend ähnliche derartige **Last-Situationen**: Die typische "Lastspitze", wochentags ≈ 10-12 Uhr und ≈ 13.30-15.30 Uhr, die typische "Nacht-Last", "Wochenend-Last", etc. Und die Beobachtung einer job-Verweilzeit v_i "bei Vormittags-Spitzenlast" oder "bei Wochenend-Last" macht, je für sich, schon etwas mehr Sinn. Eine Folgerung für die Vornahme von Messungen unter Real-Last können wir hier unmittelbar abzweigen: Da wir ja (vgl. "Unhandlichkeit") auch darauf aus sind, die Meßergebnisse ($v_1, v_2, \ldots, v_n$) irgendwie zu kompaktifizieren, sollten sie wohl aus der gleichen Last-Situation stammen, sollte also das Meßintervall MI inmitten einer solchen wiederkehrenden, wünschenswerterweise "gleichmäßigen" Last-Phase liegen, eben z.B. zwischen 10 und 12 Uhr. Andernfalls hätten die einzelnen v_i recht wenig miteinander zu tun.

Last-Situation

Nochmals: Lastumgebung

Selbst wenn wir diese Vorsicht walten lassen, kommen neue Unsicherheiten auf: Eine Last-Situation, wie oben beschrieben, ist ja nicht etwa präzise definierbar - sie stellt höchstens "in etwa" vergleichbare Verhältnisse dar. Für den einzelnen job i ist ein Einfluß des im präzisen Zeitpunkt seiner Initiierung herrschenden

Systemzustands, und auch der während seiner Anwesenheit auf das System zukommenden Belastung, auf seine Verweilzeit v_i sicher gegeben (und gelegentlich auch sehr groß: Denken Sie an einen job, der eine einzelne spezielle Magnetbandstation benötigt, die ihm, 1 msec vor seiner Bedarfsanmeldung, jemand anderes "weggeschnappt" hat - und für eine ganze Stunde behält; wäre er nur 1 msec früher dran gewesen ...). Man könnte versucht sein, angesichts dieser Schwierigkeiten in Perfektionismus zu flüchten und zu folgern, daß "alle Umstände" während der Messung zu berücksichtigen und daher aufzuzeichnen seien: Der minutiös festgehaltene Zustand des Systems bei Meß-Beginn; die präzisen Initiierungszeitpunkte aller jobs; die Zeitpunkte des Agierens interaktiver Benutzer ("Tastendruck"); die genauen Anforderungen jedes einzelnen jobs, jedes einzelnen interaktiven Kommandos, ... Eine Mammutaufgabe offenbar, und, beim zweiten Hinschauen, eine wenig lohnenswerte obendrein: Was hier erstrebt wäre zu erfassen (nämlich die exakten Umstände der "Last-Umgebung" eines jobs i) ist in dieser Detailliertheit ein absoluter Einzelfall, der "genau so" nie wiederkehrt. Eine isolierte Aussage also, mit wenig Potential für die angestrebte, wenigstens einigermaßen allgemeingültige Aussage über die Leistung eines RS, hier anhand des Kriteriums job-Verweilzeiten. Da sollte uns, trotz mangelnder Präzision, die Aussage über die Verweilzeit "bei typischer spätmorgendlicher Last-Situation" doch noch willkommener sein.

deterministische Beschreibung Lastumgebung

Statistische Berück-sichtigung Lastumgebung

So weit, so gut - die erkannte Schwierigkeit, daß der nämliche job i in seiner Verweilzeit v_i von der ihn umgebenden System-Belastung beeinflußt wird, haben wir nicht aus der Welt geschafft. Wir müssen uns wohl, statt obigen perfektionistisch-deterministischen Ansatzes, eine andere Deutung dieser Tatsache zurechtlegen. Dazu folgendes Gedankenexperiment: Wenn wir unseren job i zu verschiedenen Zeitpunkten starteten, er also auf unterschiedliche Systemzustände träfe und während seiner Verweilzeit auch durch unterschiedlichen Last-Umgebungen beeinflußt würde, würden wir auch unterschiedliche Verweilzeiten v_i für diesen job messen. In der Menge dieser Verweilzeitwerte können wir aber Regelmäßigkeiten erwarten in dem Sinne, daß gewisse Werte (bzw. gewisse Wert-Intervalle) häufiger eingenommen werden als andere. Eine Charakterisierung dieser Regelmäßigkeiten ("wie häufig groß, wie häufig klein") hätte ebenfalls den Charakter der Beschreibung der "Verweilzeit des jobs i" - sie würde zwar über die einzelne Messung (einzelner job-Lauf im Gedankenexperiment) nichts aussagen, wohl aber über die Menge dieser Messungen.

Sie haben es längst erkannt: Das mathematische Modell für dieses Phänomen einer Größe, die sich wiederholt anschauen läßt, in ihrem Wert variiert, wobei allerdings Regelmäßigkeiten des Variierens beschreibbar sind, ist die **Zufallsvariable**, deren **Verteilung** die Gesetzmäßigkeit der Häufigkeiten erfaßt, mit denen sie bestimmte Werte annimmt, bzw. in bestimmten Wertintervallen liegt.

Verweilzeit: Zufallsvariable

Sei also die Verweilzeit eines jobs i, bei einer gewissen Last-Situation, beschrieben durch eine Zufallsvariable (ab jetzt: ZV) V_i. Da wir bei einer Realisierung von V_i (einem gewissen gemessenen v_i) auf einen Wert aus einem Kontinuum möglicher Werte vorbereitet sein werden, sollte V_i als kontinuierliche ZV gewählt sein. Ihre Verteilung (die Gesetzmäßigkeit ihres Schwankens) kann somit, neben der Beschreibung durch die Verteilungsfunktion $FV_i(v):=P[V_i \leq v]$, auch durch die Dichtefunktion $fV_i(v):=dFV_i(v)/dv$ erfaßt werden. Und die Aufgabe der Ermittlung der Verweilzeit job i stellt sich jetzt dar als Ermittlung der Verteilung von V_i bzw. als Ermittlung gewisser Charakteristika dieser Verteilung, so z.B. des Erwartungswertes (Mittelwertes) $E[V_i]$. Dies natürlich aus unseren Messungen!

Beschreibung: Ermittlung Verteilungs-Charakteristika

Sammeln von Beobachtungen

Wir hatten schon in Abschn. 2.1 erahnt, daß wir statistische Methoden zur Auswertung von Messungen benötigen würden - diese Ahnung ist jetzt Gewißheit geworden. Wir werden uns im folgenden Abschn. 3.2 auch mit einigen grundlegenden statistischen Methoden auseinandersetzen. Eines sollte aber schon jetzt einleuchten: Daß eine einzelne Beobachtung der Verweilzeit von job i, also eine einzelne Realisierung v_i der ZV V_i, keine hinreichende Basis zur Ermittlung der Verteilung von V_i (der Ermittlung der Gesetzmäßigkeiten des "Schwankens" von V_i) darstellen kann. Unser Gedankenexperiment (blättern wir zurück) war ja auch anders angelegt: Wir hatten job i in Gedanken nicht nur einmal "laufen lassen", sondern verschiedene Male, dabei jedes Mal seine Verweilzeit messend; hatten also nicht von einer einzelnen Realisierung v_i von V_i aus argumentiert, sondern von einer gewissen Menge von Realisierungen $v_{i1},v_{i2},\ldots,v_{im}$ aus. Wollten wir unser Gedankenexperiment in die Praxis umsetzen, hieße das, job i "immer wieder" zu starten. Aber Vorsicht: Sollte job i nennenswerte Betriebsmittelansprüche stellen, dann verändern wir damit auch die Belastung des Systems nennenswert - und zerstören unsere mühsam erarbeitete Sicht der "wiederkehrenden typischen Last-Situation" (die ja mit vielen job i Läufen anders ist als mit nur einem solchen Lauf). Als rettender Ausweg fällt Ihnen sicher ein, den in seiner Verweilzeit zu beurteilenden job i zwar mehrmals zu starten, aber in verschiedenen Meßintervallen; bei Interesse an der "spätmorgendlichen Last-Situation" also heute um 11 Uhr, morgen um 11 Uhr, übermorgen ...; da job i ja Teil dieser Last-Situation ist, hat er morgen, übermorgen den gleichen legitimen Einfluß auf das Betriebsgeschehen wie heute! Zusätzlichen Aufwand bedeutete das allemal - wesentlicher allerdings: Unseren ursprünglichen Meß-Ansatz (in einem einzigen Meß-Intervall die Verweilzeiten aller jobs aufzuzeichnen) müßten wir schlicht vergessen - er scheint überhaupt nicht mehr zu taugen!

Realisierungs-Möglichkeiten

"job" als statistische Größe

Lehnen wir uns also zurück und bedenken unsere Aufgabe nochmals von vorne: Es ging uns um die Messung von job-Verweilzeiten bei einer gewissen Last-Situation. Daß der einzelne job, aufgrund der nicht deterministisch festzustellenden Last-Umgebung, in gewissem Rahmen schwankende Verweilzeiten aufweisen konnte, haben wir aufgefangen, indem wir diese Verweilzeit als ZV auffaßten. Daß unterschiedliche jobs (und n solche erfaßt unser Meß-Ansatz) unterschiedliche Verweilzeiten aufweisen werden, läßt sich nicht leugnen (auch wenn jede dieser Verweilzeiten durch ihre eigene ZV beschrieben ist - es werden eben unterschiedliche ZV V_i, $i=1,2,\ldots,n$ sein). Die Verweilzeiten unterschiedlicher jobs sind verschieden, weil jeder sein spezifisches Aufgabenpensum zu erfüllen hat; wir hatten in Kap. 1 diese Verschiedenheiten auch schon schematisch zu notieren versucht in Form unterschiedlicher "Verhaltensmuster" (lesen Sie nach im Bereich der Abb. 1.14/1.16). Der Versuch, für einen bestimmten job i (und sein zugeordnetes Verhaltensmuster) die Verweilzeit zu bestimmen, hatte von unserem Meß-Ansatz weggeführt - dies war der Grund für unsere jetzigen Überlegungen. Mehr noch, wenn wir uns darauf einließen, für unterschiedliche (einige?, alle?, alle denkbaren?) job-Verhaltensmuster die Verteilungen der ihnen zugeordneten Verweilzeit-ZV zu bestimmen, was hätten wir gewonnen? Eine große Menge von Verweilzeit-ZV-Charakterisierungen, untereinander nicht recht vergleichbar wegen der unendlichen Vielfalt potentieller Verhaltensmuster (RS sind programmierbar, und jedes potentielle Programm ist ein Verhaltensmuster!) und der resultierenden unendlichen Vielfalt von Möglichkeiten für jobs, "verschieden" zu sein.

Sie mögen (und sollen!) sich erinnert sehen an die Schwierigkeiten, die uns die Betrachtung der (einen job i beeinflussenden) Last-Umgebung machte - und an den mittlerweile gefundenen Ausweg: Die Vielfalt dieser deterministisch nicht faßbaren Einflüsse als "zufällig" zu akzeptieren und in ihrer Wirkung auf die Verweilzeit eines jobs dadurch zu beschreiben, daß diese Verweilzeit als ZV betrachtet wurde. Hilft uns die Statistik auch bei der neuerdings aufgetretenen Schwierigkeit der großen Vielfalt unterschiedlicher jobs? Nun, wir können folgendermaßen argumentieren: Wir betrachten das System in einer irgendwie gleichmäßigen und auch wiederkehrenden Last-Situation. Der Begriff Last-Situation (die in bestimmten Betriebs-Intervallen typischerweise auftretende Belastung) bezeichnet für diese Situation charakteristische Arten einzelner System-Beauftragungen und die Regelmäßigkeiten (z.B. Häufigkeiten) ihres Auftretens. So gesehen, sind die verschiedenen jobs, die in einem Meß-Intervall beobachtet werden, Bestandteil (einer möglichen Ausprägung) der herrschenden Last-Situation. Und weiter: Jeder einzelne job ist, wenn wir seinen Besonderheiten nicht deterministisch-präzise nachgehen wollen, ein "zufällig" herausgegriffenes Exemplar aus der Menge der für die gegebene Last-Situation charakteristischen jobs/job-Arten. Noch ein kleiner Gedankensprung: Wir können unsere ursprüngliche Messung, resultierend in der Menge/Folge von job-Verweilzeiten eines Meß-Intervalls (aus der Mitte einer "regelmäßigen" Last-Situation)

(3.1.2) $(v_1, v_2, \ldots, v_n)$

prototypischer job

auffassen als Folge von n Realisierungen **einer** ZV V, als Realisierungen **der** zufälligen Verweilzeit **des** für diese Last-Situation prototypischen jobs, wobei die Werteschwankungen nun gleichzeitig den Einfluß der zufälligen Auswahl eines konkreten jobs **und** den Einfluß der zufälligen Last-Umgebung zu erklären haben. Die Fiktion dieses einen, Last-Situation-spezifischen, zufälligen jobs mag Ihnen vielleicht allzu grobschlächtig vorkommen. Sie ist aber einerseits eine taugliche Basis für eine von den eingangs beklagten Unsicherheiten nicht mehr belastete Interpretation der Meßwerte (3.1.2); sie läßt sich andererseits leicht verfeinern, wobei aber die ausgeführte Gedankenkette weiterhin erhalten bleibt.

Testfrage 3.1.3: Angenommen, die jobs aus Ihrem Meßintervall sind sehr "verschieden", so daß Sie sie nicht "in einen Topf" werfen mögen, sie nicht durch eine einzige Verweilzeit-ZV V repräsentiert sehen wollen. Welche Möglichkeiten sehen Sie, über verschiedene "job-Klassen" zu verfeinerten Beobachtungen zu kommen? Sehen Sie auf dieser Basis eine Chance, aus dem zweifachen statistischen Ansatz (Unsicherheit der Last-Umgebung, Unsicherheit bzgl. konkretem job-Exemplar) herauszukommen?

Testfrage 3.1.4: Verfolgen Sie die am Beispiel "Verweilzeiten" vorgeführte Gedankenkette an der zweiten der eingangs vorgeschlagenen Konkretisierungen, dem Beispiel der "CPU-busy periods", bis hin zu einer ähnlich vollständigen Übersicht über das diesbezügliche Meß- bzw. Meßinterpretations-Problem.

Diskussion Reallast

Nun noch ein wenig Statistik und wir sind fertig!? Leider nein - ich muß nochmals Salz in die Suppe schütten: Wir hatten zu Anfang von Abschn. 3.1 drei mögliche Fragestellungen (i,ii,iii) wiederholt. Gewappnet sind wir bis jetzt nur für (i)! Überlegen wir: Für (ii,iii) wird es darauf ankommen, mehrere Experimente zu machen. Eines (zu Vergleichszwecken), um den "jetzigen Status" festzustellen, also entsprechend (i); zumindest ein weiteres (nach vollzogener Veränderung etwa der Maschine), um das Ausmaß der resultierenden Leistungsveränderung zu beurteilen. Wir sind inzwischen vorsichtig genug, jeweils bei fester Last-Situation zu arbeiten. Also: Status-Feststellung "heute" (z.B.:) zwischen 10 und 12 Uhr, Veränderungs-Feststellung (nach Maschinen-Änderung) "morgen" zwischen 10 und 12 Uhr. Beides unter Real-Last, nach dzt. Wissensstand. Machen wir unsere Leistungs-Beurteilung wieder an den beobachteten job-Verweilzeiten fest und überlegen weiter: Würden wir keine Maschinen-Veränderung vornehmen, wären wir "morgen" auf andere Verweilzeit-Werte gefaßt als "heute"; klarerweise, denn wir üben ja keinerlei Einfluß auf die lasterzeugenden Benutzer aus; ihr Verhalten ist zwar irgendwie ähnlich (gleiche Last-Situation), aber doch nicht identisch (gerade um dieses Faktum zu erfassen, hatten wir ja zu dem ZV-Modell für Verweilzeiten gegriffen!). Wenn wir jetzt die Maschinen-Veränderung vornehmen, erwarten wir "morgen" nochmals, über diese erwarteten Schwankungen hinausgehend, veränderte Verweilzeit-Werte. Wir haben aber nur die gemessenen Verweilzeiten selbst in der Hand - sie sind "anders" aus nunmehr zwei Gründen (statistisches Schwanken innerhalb Last-Situation, Einfluß Maschinen-Veränderung), und es gälte, diese beiden Ursachen anhand ihrer kumulativen Wirkung (andere Verweilzeiten) "auseinanderzudividieren", sind wir doch nur am Einfluß der Maschinen-Veränderung interessiert. Es gibt auch durchaus statistische Verfahren, solches im Prinzip durchzuführen (Hypothesentest: "signifikante" Veränderung), doch sei hier verraten, daß diese Ansätze nur bei wirklich drastischen Eingriffen in die Maschine und hervorgerufenen drastischen Einflüssen auf die Verweilzeiten zufriedenstellend funktionieren und sich bei den (weitaus häufigeren) "leichteren" Eingriffen oft als nicht hinreichend trennscharf erweisen.

Kritik

Nicht verzagen, der Ausweg aus dieser neuerlichen Schwierigkeit liegt auf der Hand: Leistung hängt von Maschine und Last ab; wenn wir z.B. die Maschine verändern, sollten wir die resultierende Leistungsveränderung bei gleicher Last studieren. Sind uns zwei Ausprägungen der gleichen Last-Situation (heute 10-12, morgen 10-12) nicht "gleich genug", müssen wir dafür sorgen, daß sie "gleicher" werden. Dies bedingt aber, daß wir Einfluß auf die Belastung nehmen müssen - bedingt einen Abschied von den Messungen unter Real-Last (eine beeinflußte Last ist nicht mehr "real" in dem Sinne, daß sie "von alleine geschähe", aus dem Agieren der Benutzer selbst heraus). Wir beschreiben damit einen Weg

- weg vom **unkontrollierten Experiment**, unter nicht beeinflußter Real-Last
- hin zum **kontrollierten Experiment**, bei genau geplanter, eigens aufgebrachter, mehr oder weniger "künstlicher" Last.

kontrolliertes Experiment

Die damit auftretenden Schwierigkeiten praktischer Art sind offensichtlich: Es geht ja darum, (zumindest) zweimal die gleiche Last auf die Maschine zu bringen. Klar, die Maschine (das RS) muß während der Abarbeitung dieser Last und den dabei erfolgenden Messungen aus dem Normalbetrieb ausgegliedert werden, wir müssen Experimente im **Exklusivbetrieb** fahren. Weiterhin ist die exakte Aufbringung (und nachfolgend nochmalige Reproduktion) einer Last keine einfache Aufgabe. Wir können diese Aufgabe zum einen dadurch angehen, daß wir uns Möglichkeiten verschaffen, eine deterministisch präzise beschriebene Belastung (eine feste Menge "lauffähiger" Aufträge samt Zeitpunkten der Beauftragung) zu reproduzieren; dies bringt uns zurück zu den schon breit diskutierten Schwierigkeiten deterministisch-präziser Last-Beschreibungen, aus denen wir ja mittels der Idee "Last-Situation" zu fliehen versucht hatten; dies involviert weiterhin zu schaffende Mechanismen zur Initiierung bestimmter Aufträge zu bestimmten Zeiten - eine zusätzliche Instrumentierung zur Last-Aufbringung. Zum anderen könnten wir diese Aufgabe dadurch anzugehen versuchen, daß wir die zu reproduzierende Last nicht in deterministischem Sinne "gleich" machen, sondern in statistischem Sinne (wo uns doch die Statistik schon einige Male geholfen

Exklusivbetrieb

hatte!). Von letzterem Ansatz könnten wir uns Vorteile im Hinblick auf eine "sparsamere" Last-Beschreibung erhoffen; die Notwendigkeit eines Mechanismus' zur Last-Aufbringung bleibt aber sicher bestehen.

repräsentative Last

Fazit: Um Objekt-Experimente zweifelsfrei interpretieren ("bewerten") zu können, sind uns kontrollierte Experimente willkommener als unkontrollierte. Kontrollierte Experimente bedürfen genauer Beschreibungen der zu verwendenden Last, wobei die Beschreibungen "das Typische" einer gewählten Last-Situation zu erfassen hätten, in der Fachterminologie: **repräsentativ für eine Last-Situation** sein müßten. Um eine repräsentative, reproduzierbare Last müssen wir uns aber sicher eigens bemühen; sie kann zunächst nur aus Beobachtungen realer Betriebsintervalle (Ausprägungen der fraglichen Last-Situation) abgeleitet werden.

Messungen von Leistung und Last

Beobachtungen aber sind ja Messungen (jetzt: der Belastung); wo wir die Rolle von Messungen bisher allein im Bereich der Beobachtung von Leistungskriterien sahen (bzw. von Größen, aus denen Leistungsmaße ableitbar waren), erkennen wir jetzt, daß Messungen auch eine zentrale Rolle bei der Vorbereitung kontrollierter Experimente zu übernehmen haben, nämlich bei der Beobachtung von Lasten zum Zwecke der repräsentativen Beschreibung interessierender Last-Situationen.

3.2 Ein wenig Statistik

Zufallsvariable: Modell

Betrachten wir eine eindimensionale ZV Y als mathematisches Modell einer numerischen Größe,

- die wir in der Realität wiederholt beobachten können ("Experimente");
- deren Wert von Beobachtung zu Beobachtung (potentiell) variiert;
- bei der wir aber gewisse Regelmäßigkeiten des Variierens erwarten in dem Sinne, daß bei Vorliegen einer gewissen Menge dieser Beobachtungen (nach Durchführung einer Menge von Experimenten) die beobachteten Werte mit bestimmten relativen Häufigkeiten in bestimmten Werte-Intervallen liegen.

Verteilung

Die Gesetzmäßigkeiten des Variierens sind auf seiten der (Modell-) **Zufallsvariablen** Y erfaßt durch deren **Verteilung**, welche für jedes Werte-Intervall I=[u,o] festlegt, mit welcher **Wahrscheinlichkeit** P[Y I] eine (jede) **Realisierung** von Y in diesem Intervall liegt. In statistischer Interpretation dienen diese Intervall-Wahrscheinlichkeiten als Modell der (erwarteten) relativen Häufigkeiten, mit denen die Beobachtungen aus einer Menge von Experimenten in eben jene Intervalle zu liegen kommen.

Charakterisierung:

Es gibt eine ganze Reihe von Möglichkeiten, die Verteilung einer ZV Y vollständig und doch sparsam zu charakterisieren. Sie kennen sicher die **Verteilungsfunktion**

Verteilungsfunktion

(3.2.1) $FY(y) := P[Y \leq y]$

welche für alle festen Werte y angibt, mit welcher Wahrscheinlichkeit Realisierungen von Y kleiner oder gleich y sind (in der Interpretation: mit welcher relativen Häufigkeit Beobachtungen aus einer Menge von Experimenten im Intervall $(-\infty, y]$ zu erwarten sind). Bei kontinuierlichen ZV (deren mögliche Realisierungen ein Kontinuum von Werten, ein kontinuierliches Intervall, überdecken) stellt die **Verteilungsdichte**(-funktion) fY(y) ebenfalls eine vollständige Charakterisierung der Verteilung dar. Sie hängt mit der Verteilungsfunktion bekanntlich gemäß

Verteilungsdichte

(3.2.2a) $fY(y) = dFY(y) / dy$

(3.2.2b) $FY(y) = \int_{-\infty}^{y} fY(y')\, dy'$

zusammen. Aus beiden Charakterisierungen lassen sich z.B. die zuvor einführend verwendeten Intervallwahrscheinlichkeiten zu

(3.2.3a) $P[Y \ \ (u,o)] = \int_{u}^{o} fY(y)\, dy$

(3.2.3b) $P[Y \ \ (u,o)] = FY(o) - FY(u)$

bestimmen. Anstelle solch vollständiger Charakterisierungen einer Verteilung (wie gesagt, es gibt deren noch eine ganze Reihe) begnügt man sich häufig mit partiellen Charakterisierungen. Zu ihnen gehören die sog. **Momente** einer ZV (bzw. ihrer Verteilung); dabei ist (zur Erinnerung) das k-te Moment μ_k als Erwartungswert $E[Y^k]$ definiert, d.h. zu

Momente

(3.2.4) $\mu_k := E[Y^k] = \int_{-\infty}^{+\infty} y^k\, fY(y)\, dy \qquad k=1,2,..$

Zwar charakterisiert die Menge **aller** Momente (sofern diese endlich sind, "existieren") die Verteilung ebenfalls vollständig - man begnügt sich aber eben häufig mit einigen wenigen, die dann den Charakter der Verteilung lediglich "so ungefähr" beschreiben. Das im Sinne der Charakterisierung einer Verteilung wohl interessanteste (einzelne) Moment ist das erste, der Erwartungswert der ZV Y selbst, der "zentrale" Wert von Y in dem Sinne, daß er die horizontale Lage des Schwerpunkts der von Abszisse und Verteilungsdichte berandeten Fläche anzeigt (vgl. (3.2.4) für k=1). Entsprechend erlauben auch die höheren Momente (k>1) Aussagen über Eigenschaften der Verteilung, charakterisieren sie also genauer (immer wieder im Hinblick auf die Form des Graphen der Verteilungsdichte); so beinhaltet das zweite Moment Aussagen über die "Breite" der Verteilung, das dritte über deren "Symmetrie", das vierte über deren "Spitzigkeit". Eine genauere Charakterisierung dieser Eigenschaften wird dabei allerdings mittels aus diesen Momenten ableitbaren speziellen Kenngrößen unternommen. So ist die Breite der Verteilung einer ZV Y charakterisierbar durch ihre **Varianz** V[Y], das zweite Zentralmoment, wobei k-te **Zentralmomente** definiert sind als

Form Verteilungsdichte

Varianz

(3.2.5a) $E[(Y-E[Y])^k] \qquad k=2,3,\ldots$

und speziell die Varianz (k=2)

(3.2.5b) $V[Y]:= E[(Y-E[Y])^2]$

Streuung

Die positive Wurzel der Varianz trägt den Namen **Streuung**

(3.2.5c) $\sigma := +\sqrt{V[Y]}$

und eine noch bessere, relative Breiten-Charakterisierung

Variations-Koeffizient

(3.2.5d) $\nu := \sigma/\mu_1$

die Bezeichnung Variationskoeffizient.

Testfrage 3.2.6: Wie ist der genaue Zusammenhang zwischen erstem und zweitem Moment (μ_1,μ_2) und Streuung (V[Y]) einer ZV Y? Wenn Sie sich nicht erinnern: Werten Sie einfach Definitionsgleichung (3.2.5b) aus.

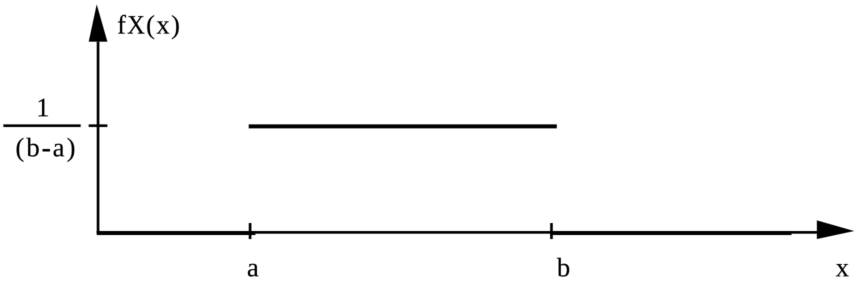
Zur Stützung der Vorstellungskraft:

Gleichverteilung

Beispiel 3.2.7: Eine sehr einfache Verteilung ist die sog. **Gleichverteilung** über einem Intervall [a,b], a<b. Die Dichtefunktion einer gleichverteilten ZV X lautet

$$fX(x) = \begin{cases} 0 & x<a \\ 1/(b-a) & a\le x\le b \\ 0 & x>b \end{cases}$$

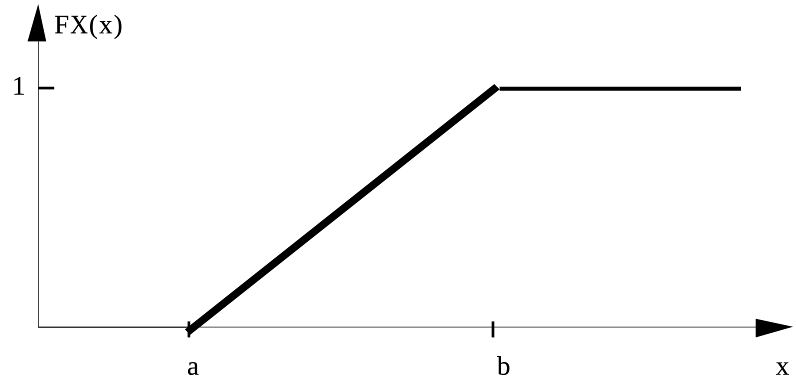
Der Graph der Dichte hat demzufolge das Aussehen



d.h. eine Realisierung von X fällt mit gleicher Wahrscheinlichkeit in (beliebige) gleichbreite Intervalle des Wertebereichs [a,b]. Der Vollständigkeit halber die Verteilungsfunktion

$$FX(x) = \begin{cases} 0 & x<a \\ (x-a)/(b-a) & a \le x \le b \\ 1 & x>b \end{cases}$$

und ihr Graph



Der Erwartungswert von X (das erste Moment) ergibt sich zu

$$\mu_1 = \int_a^b \frac{x}{b-a}\,dx = \frac{b+a}{2}$$

das zweite Moment zu

$$\mu_2 = \int_a^b \frac{x^2}{b-a}\,dx = \frac{b^2+a\,b+a^2}{3}$$

die Varianz zu

$$V[X] = \mu_2 - \mu_1^2 = (b-a)^2 / 12$$

die Streuung zu

$$\sigma = \frac{b-a}{\sqrt{12}}$$

der Variationskoeffizient zu

$$\nu = \frac{b-a}{\sqrt{3}\,(b+a)}$$

Also: Der "zentrale" Wert liegt, wo man ihn erwartet; die Breitemaße sind der Verteilungsform entsprechend relativ groß (sie werden kleiner, wenn die Verteilung sich mehr um μ_1 konzentriert, größer, wenn die Verteilung mehr nach links und rechts hinausreicht; spielen Sie in Gedanken mit Intervallgrenzen a und b).

Ermittlung Charakteristika

So viel als Hintergrund. Unsere konkrete Situation (vgl. Einleitung dieses Abschnitts) besteht daraus, daß wir eine "variierende" Größe wiederholt beobachten können, so daß schließlich eine Menge von Beobachtungen $\{y_1,y_2,\ldots,y_n\}$ vorliegt; daß wir gerne aus diesen Beobachtungen Eigenschaften der (gemäß unserer Vorstellung) zugeordneten (Modell-)ZV Y ableiten möchten bzw. konkreter: Eigenschaften der Verteilung von Y (diese hat ja die Regelmäßigkeiten des Variierens der Beobachtungsgröße zu erfassen). Man nennt die vorliegende Beobachtungsmenge $\{y_1,y_2,\ldots,y_n\}$ eine **Stichprobe**. Wir wollen vereinbaren, daß die Beobachtungen so ausgeführt werden, daß die Stichprobe **unabhängig** ist; will heißen, daß mit Kenntnis von m(<n) dieser Beobachtungswerte nicht mehr über die restlichen n-m Werte ausgesagt werden kann als ohne diese Kenntnis.

Stichprobe

Unabhängigkeit

Testfrage 3.2.8: Machen Sie sich den Begriff "unabhängige Stichprobe" klar anhand folgenden Beispiels: Jemand erzählt Ihnen, er habe 10-mal gewürfelt; er habe 9-mal die "sechs" erzielt; er fragt nach Ihrer Erwartung bzgl. des 10-ten Wurfs. Antworten Sie unter Benutzung einer Argumentation über die Unabhängigkeit (oder Abhängigkeit?) der Stichprobe im Rahmen zweier verschiedener Szenarien: Im einen Szenario ist Ihnen bekannt, daß die Stichprobe aus dem simultanen Werfen von 10 Würfeln stammt; im anderen wissen Sie, daß nur ein Würfel (10-mal) benutzt wurde. Es handelt sich um nicht manipulierte ("ideale") Würfel.

Diskussion unabhängige Stichprobe

Betrachten wir die Unabhängigkeit einer Stichprobe noch von einer anderen Seite: Angenommen, die y-Werte einer Stichprobe seien einer nach dem anderen ermittelt worden (oder: Sie griffen aus einer vorliegenden Stichprobe einen y-Wert nach dem anderen heraus), so daß sich die Stichprobe auch als Folge von Werten $(y_1,y_2,\ldots,y_n)$ darstellt. Wir haben Wiederholbarkeit unserer Beobachtungen vorausgesetzt, können also eine weitere Stichprobe (wieder aus n Werten) ziehen. Wir setzen das Spiel fort bis (z.B.) k Stichproben vorliegen. Mit leichter Umbenennung der beobachteten Werte verfügen wir über die Beobachtungen

Stichprobe 1: $y_{11},y_{12},\ldots,y_{1n}$
Stichprobe 2: $y_{21},y_{22},\ldots,y_{2n}$
.
.
Stichprobe k: $y_{k1},y_{k2},\ldots,y_{kn}$

Da jede vorgenommene Beobachtung der gleichen Regelmäßigkeit des Variierens folgt (beschrieben durch die Verteilung von Y), erwarten wir innerhalb einer Stichprobe, daß die relativen Häufigkeiten des Einnehmens bestimmter Werte-Intervalle dieser Regelmäßigkeit folgen (also entsprechend Y-Verteilung ausfallen); genauer: wir erwarten dies für jede der Stichproben 1 bis k. Mit der gleichen Argumentation erwarten wir aber, quer zu den Stichproben, für die Werte an jeweils fester (erster bis n-ter) Position die nämlichen relativen Häufigkeiten, erwarten also, daß die Beobachtungsmenge $\{y_{11},y_{21},\ldots,y_{k1}\}$ (ebenso wie die Beobachtungsmengen an zweiter, …,n-ter Position) die gleiche Regelmäßigkeit des Variierens aufweist. Wir haben damit eine etwas andere Vorstellung vom Wesen einer Stichprobe erarbeitet: An jeder Position sind wir auf Variieren gefaßt, beschreiben die Beobachtung auf i-ter Position durch "ihre" ZV Y_i, haben es demnach mit n ZV zu tun, insgesamt mit der Folge von ZV

(3.2.9) $(Y_1,Y_2,\ldots,Y_n)$

Alle ZV Y_i gehorchen derselben Verteilung, nämlich jener von Y: Die Y_i sind **identisch verteilt**. Eine präzise Definition der Unabhängigkeit der Stichprobe ist, daß die ZV $Y_1,Y_2\ldots Y_n$ unabhängig verteilt sind, daß also (Sie erinnern sich:) ihre gemeinsame Verteilungsfunktion dem Produkt der individuellen (Rand-)Verteilungsfunktionen entspricht, in Zeichen

Unabhängigkeit präzisiert

$$P[Y_1\le y_1,Y_2\le y_2,\ldots,Y_n\le y_n] = P[Y_1\le y_1]\cdot P[Y_2\le y_2]\cdot\ldots\cdot P[Y_n\le y_n]$$

bzw., da alle Y_i identisch (und zwar wie Y) verteilt sind

$$= FY(y_1)\cdot FY(y_2)\cdot\ldots\cdot FY(y_n)$$

Zurück zu der einzelnen (unabhängigen) Stichprobe $(y_1,y_2,\ldots,y_n)$, aus der wir Charakteristika der Y-Verteilung ermitteln wollen. Fangen wir bescheiden mit der Ermittlung des ersten Moments μ_1=E[Y] an, das wir auch "zentraler Wert" der Verteilung nannten. Steht dieser zentrale Wert zum Mittelwert der Stichprobe, definiert als

(3.2.10) $$m := \frac{1}{n} \sum_1^n y_i$$

in irgendeiner Verbindung? Ein Indiz dafür ist das (Ihnen sicher erinnerliche!) **Gesetz der großen Zahlen**, nach dem gilt, daß

Gesetz der großen Zahlen

(3.2.11) $$\lim_{n\to\infty} P\left[\left|\frac{1}{n} \sum_1^n Y_i - E[Y]\right| \ge \varepsilon\right] = 0$$

für jedes $\varepsilon>0$ (bei identisch wie Y verteilten Y_i); in Worten, daß $\Sigma Y_i / n$ stochastisch gegen E[Y] konvergiert.

Machen wir uns das ganz klar: Da wir im Prinzip verschiedene (beliebig viele) Stichproben ziehen können, im praktischen Leben alle endlich mit (z.B.) Umfang n, und jede Stichprobe "etwas anders" ausfällt, werden die Stichprobensummen (wieder Doppelindizierung der Beobachtungen)

$$\sum_i y_{ji} \qquad j = 1,2,\ldots$$

der diversen Stichproben j ebenfalls in bestimmter Art variieren. Klar doch, da jede Position i der Stichprobe durch eine ZV Y_i erfaßt war, ist ihre Summe

$$S_n := \sum_{i=1}^n Y_i$$

ebenfalls eine ZV (mit ihrer eigenen Verteilung). Das Gesetz der großen Zahlen besagt nun, daß die Verteilung von S_n / n (ebenfalls eine ZV) sich mit wachsendem n immer stärker um den Wert E[Y] konzentriert; daß die Wahrscheinlichkeit, eine Realisierung von S_n / n zu erhalten, die um mehr als irgendein ε von E[Y] entfernt liegt, mit wachsendem n gegen 0 strebt (wobei man ε auch noch beliebig klein wählen kann!).

Somit haben wir festzuhalten: Der Mittelwert (3.2.10) einer Stichprobe kann nicht **das** gesuchte erste Moment liefern, er variiert ja von Stichprobe zu Stichprobe, ist **eine** Realisierung der ZV S_n / n. Die Realisierung sollte aber mit hoher Wahrscheinlichkeit "nahe" am interessierenden Wert μ_1 liegen (zumindest für hinreichend großes n) - das verriet uns das Gesetz der großen Zahlen und liefert damit

Mittelwert ist Zufallsvariable:

Schätzwert
Schätzer

die Berechtigung, mit dem Stichproben-Mittelwert m gemäß (3.2.10) als **Schätzwert** für μ_1 zu arbeiten. m ist dabei Realisierung einer ZV, des sog. **Schätzers** S_n / n für μ_1. Um die verschiedenen Begriffe klar auseinander zu halten, folgende Vereinbarung für Bezeichnungen:

Zielgröße

• Wir wollen ermitteln das erste Moment

$$\mu_1$$

Schätzer

• Wir kennen einen hoffnungsvollen Schätzer für μ_1, die ZV

(3.2.12a) $$\tilde{\mu}_1 := \frac{1}{n} \sum_{i=1}^{n} Y_i$$

Schätzwert

• Eine einzelne Realisierung der Schätzvariablen (einen Schätzwert für μ_1) erhalten wir aus den Stichprobenwerten zu

(3.2.12b) $$\mu_1^* := \frac{1}{n} \sum_{i=1}^{n} y_i$$

Da wir erkannt haben, daß ein Schätzer eine ZV ist, die naheliegende Frage nach Charakteristika **dieser** (Schätz-)ZV. Ihr erstes Moment (Hinweis auf Zentrum der Verteilung):

$$E[\tilde{\mu}_1] = E[(\sum Y_i) / n] \qquad \text{s. (3.2.12a)}$$

Ich setze die "Rechenregeln" für Erwartungswerte (linearer Operator), also

$$E[a \cdot X + b \cdot Z] = a \cdot E[X] + b \cdot E[Z]$$

als bekannt voraus. Wir erhalten damit weiter:

$$E[\tilde{\mu}_1] = \frac{1}{n} \sum_{i=1}^{n} E[Y_i]$$

Alle Y_i sind identisch wie Y verteilt:

$$= \frac{1}{n} \sum_{i=1}^{n} E[Y]$$

(3.2.13) $$E[\tilde{\mu}_1] = \mu_1 \qquad \text{vgl. (3.2.4)}$$

Erwartungs-Treue

Dies ist ein willkommenes Ergebnis, besagt es doch, daß auch bei endlichem Stichprobenumfang ($n<\infty$, beliebig) der Erwartungswert des Schätzers $\tilde{\mu}_1$ mit der zu schätzenden Größe μ_1 übereinstimmt (daß also die Verteilung von $\tilde{\mu}_1$ als Zentrum den Wert μ_1 aufweist). Man nennt einen Schätzer mit dieser Eigenschaft **erwartungstreu**.

Testfrage 3.2.14: Zeigen Sie, daß die ZV

$$\tilde{\mu}_k := \frac{1}{n} \sum_{1}^{n} Y_i^k \qquad k=2,3,...$$

erwartungstreue Schätzer der k-ten Momente μ_k der Verteilung von Y sind.

Um Sie vor Trugschlüssen zu bewahren: Es ist **nicht** selbstverständlich, daß ein aus irgendeinem Grunde "hoffnungsvoller" Schätzer auch erwartungstreu ist! Wir könnten z.B., in Analogie zur Formel (3.2.12a) für den $\tilde{\mu}_1$-Schätzer, für die Varianz von Y (vgl. (3.2.5b,c)) den Schätzer

Schätzung von σ^2

$$\tilde{\sigma}_a^2 := \frac{1}{n} \sum_1^n (Y_i - E[Y])^2$$

vorschlagen und, da wir ja E[Y] nicht kennen, stattdessen den zugehörigen Schätzer $\tilde{\mu}_1$ einsetzen, also insgesamt

$$\tilde{\sigma}_b^2 := \frac{1}{n} \sum_1^n \left(Y_i - \tilde{\mu}_1\right)^2$$

verwenden. Fragen wir uns, ob $\tilde{\sigma}_b{}^2$ erwartungstreu ist:

$$E\left[\tilde{\sigma}_b^2\right] = E\left[\frac{1}{n} \sum_1^n \left(Y_i - \tilde{\mu}_1\right)^2\right]$$

mit (3.2.12a) und der Linearität des E-Operators:

$$E\left[\tilde{\sigma}_b^2\right] = \frac{1}{n} \sum_{i=1}^n E\left[\left(Y_i - \frac{1}{n} \sum_{j=1}^n Y_j\right)^2\right] = \frac{1}{n^3} \sum_{i=1}^n E\left[\left(n\, Y_i - \sum_{j=1}^n Y_j\right)^2\right]$$

Eine Umformung des Erwartungswertarguments liefert:

$$\left(n\, Y_i - \sum_{j=1}^n Y_j\right) = n \left(Y_i - \mu_1\right) - \sum_{j=1}^n \left(Y_j - \mu_1\right)$$

$$(\ldots)^2 = n^2 \left(Y_i - \mu_1\right)^2 - 2\, n \left(Y_i - \mu_1\right) \sum_{j=1}^n \left(Y_j - \mu_1\right) + \left(\sum_{j=1}^n \left(Y_j - \mu_1\right)\right)^2$$

Damit wird aus dem i-Summanden obiger Summe:

$$E\left[(\ldots)^2\right] = n^2\, E\left[\left(Y_i - \mu_1\right)^2\right] - 2\, n\, E\left[\left(Y_i - \mu_1\right)^2\right] - 2\, n \sum_{j=1, \neq i}^n E\left[\left(Y_i - \mu_1\right) \left(Y_j - \mu_1\right)\right]$$
$$+ \sum_{j=1}^n E\left[\left(Y_j - \mu_1\right)^2\right] + \sum_{j=1}^n \sum_{k=1, \neq j}^n E\left[\left(Y_i - \mu_1\right) \left(Y_k - \mu_1\right)\right]$$

sowie mit (3.2.5b,c) und der Identität aller Y_i-Verteilungen:

$$E\left[(\ldots)^2\right] = n^2 \sigma^2 - 2\, n\, \sigma^2 - 2\, n \sum \ldots \quad + n\, \sigma^2 + \sum \sum \ldots$$

Gegeben zwei ZV X_1 und X_2 heißt der Ausdruck

$$E[(X_1 - E[X_1]) \cdot (X_2 - E[X_2])]$$

Kovarianz

Kovarianz von X_1 und X_2. Sind X_1, X_2 unabhängige ZV, dann verschwindet ihre Kovarianz. Davon können Sie sich leicht überzeugen, wenn Sie von der Definition der Unabhängigkeit ausgehend (gemeinsame Verteilungsfunktion ist Produkt der Rand-Verteilungsfunktionen, gemeinsame Dichte ist Produkt der Rand-Dichten) ermitteln, daß $E[X_1 \cdot X_2] = E[X_1] \cdot E[X_2]$ ist, und mit diesem Ergebnis obige Definitionsgleichung der Kovarianz auswerten. Vorsicht: Es gilt (umgekehrt) nicht, daß aus verschwindender Kovarianz Unabhängigkeit folgt! Die Summanden unter obigen Summen sind, mit dieser Definition, Kovarianzen je verschiedener Y_i, Y_j. Mit unserer Voraussetzung der unabhängigen Stichprobe (Sie bemerken, daß dies wichtig war) verschwinden diese Summanden und damit die ganzen Summen, und wir erhalten für den bearbeiteten i-Summanden

$$E\left[\left(\ \ldots\right)^2\right] = n^2\,\sigma^2 - 2\,n\,\sigma^2 + n\,\sigma^2$$

Setzen wir dieses Zwischenergebnis in den Gesamtausdruck ein, dann ergibt sich:

$$E\left[\tilde{\sigma}_b^2\right] = \frac{1}{n^3}\; n\left\{n^2\,\sigma^2 - n\,\sigma^2\right\}$$
$$= \frac{n-1}{n}\;\sigma^2$$

asymptotisch erwartungstreu

$\tilde{\sigma}_b{}^2$ ist also **nicht** erwartungstreu. Wohl aber **asymptotisch erwartungstreu,** wofür die diesbezügliche Definition

$$\lim_{n \to \infty} E\left[\tilde{\sigma}_b^2\right] = \sigma^2$$

offensichtlich zutrifft. Leicht können wir aus $\tilde{\sigma}_b$ aber einen erwartungstreuen Schätzer $\tilde{\sigma}$ machen mittels

$$\tilde{\sigma}^2 = \frac{n}{n-1}\;\tilde{\sigma}_b^2$$

erwartungs-treuer Varianz-Schätzer

Diesen erwartungstreuen Schätzer für die Varianz von Y

(3.2.15) $$\tilde{\sigma}^2 := \frac{1}{n-1}\sum_1^n\left(Y_i - \tilde{\mu}_1\right)^2$$

wollen wir ab jetzt verwenden, wo immer wir ihn brauchen.

Punktschätzer

In unserer Erörterung der Möglichkeiten, aus unabhängigen Stichproben einer (als ZV modellierten) variierenden Beobachtungsgröße verschiedene Charakteristika der zugeordneten ZV Y abzuleiten, haben wir Schätzer für ihre k-ten Momente und für ihre Varianz kennengelernt. Man nennt diese Schätzer **Punktschätzer**, da sie jeweils nur eine Realisierung (einen "Punkt") der als ZV erkannten Schätzgrößen liefern. Diese Schätzer waren alle erwartungstreu, ihre Verteilung wies also das richtige Zentrum auf. Dennoch variieren aber die konkret errechneten Schätzwerte, z.B. μ_1*, um dieses Zentrum - ein einzelner Schätzwert kann durchaus, "unglücklicherweise", weit vom Zentrum entfernt liegen. "Wie weit" sollen/müssen wir denn befürchten? Nun, es kommt sicher auf die Breite der Verteilung der Schätz-ZV an. Versuchen wir doch diese Breite zu ermitteln anhand hierfür als geeignet bezeichneter Indikatoren, etwa der Varianz; vgl. (3.2.5b,c).

Schwankungs-Breite μ_1-Schätzer

Nehmen wir wieder unser interessantestes Charakteristikum her, den Erwartungswert μ_1 der Beobachtungsgröße bzw. ihren Schätzer $\tilde{\mu}_1$. Wie breit ist dessen Verteilung?

$$V[\tilde{\mu}_1] = E[(\tilde{\mu}_1 - E[\tilde{\mu}_1])^2]$$

mit der Erwartungstreue von μ_1 und (3.2.12):

$$V\left[\tilde{\mu}_1\right] = E\left[(\tilde{\mu}_1 - \mu_1)^2\right]$$
$$= E\left[\left(\sum Y_i/_n - \mu_1\right)^2\right]$$
$$= E\left[\left\{\frac{1}{n}\sum_{i=1}^n\left(Y_i - \mu_1\right)\right\}^2\right]$$

$$V[\tilde{\mu}_1] = \frac{1}{n^2} \left\{ \sum_i E\left[\left(Y_i-\mu_1\right)^2\right] + \sum_i \sum_{j \neq i} E\left[\left(Y_i-\mu_1\right)\left(Y_j-\mu_1\right)\right] \right\}$$

wegen verschwindender Kovarianz bei unabhängiger Stichprobe (schon wieder wesentlich!):

$$V[\tilde{\mu}_1] = \frac{1}{n^2} \sum_i E\left[\left(Y_i-\mu_1\right)^2\right]$$

wegen Identität der Y_i-Verteilungen:

$$= 1/n^2 \cdot n\sigma^2$$

(3.2.16) $$V[\tilde{\mu}_1] = \frac{\sigma^2}{n} \qquad \left(= \frac{V[Y]}{n}\right)$$

Ein Ergebnis, das voll unseren Erwartungen entspricht: Die Schwankungsbreite des Schätzers $\tilde{\mu}_1$ (ausgedrückt durch dessen Varianz) steigt mit der Schwankungsbreite der beobachteten Größe und sinkt mit dem Stichprobenumfang.

Bezogen auf das konkrete Beispiel des Erwartungswertschätzers $\tilde{\mu}_1$ gehen wir nach wie vor der Frage nach, wie weit ein berechneter Schätzwert μ^*_1 wohl vom eigentlich interessierenden "wahren" Erwartungswert μ_1 entfernt ist; bzw. präziser: Wie die Wahrscheinlichkeit (rel. Häufigkeit) einzuschätzen ist, daß $\tilde{\mu}_1$ um mehr als irgendein "tolerierbares" ε von μ_1 entfernt liegt, in Zeichen

(3.2.17) $$P[\,|\,\tilde{\mu}_1-\mu_1\,| \geq \varepsilon] = ?$$

Prinzipskizze 3.2.18 möge die vorliegende Fragestellung verdeutlichen. $\tilde{\mu}_1$ ist eine ZV, deren genaue Verteilung wir nicht kennen; die in Abb. 3.2.18 eingezeichnete Dichtefunktion von $\tilde{\mu}_1$ ist insofern nur als potentiell mögliche Form zu verstehen. $\tilde{\mu}_1$ ist ein erwartungstreuer Schätzer, so daß als Zentrum der $\tilde{\mu}_1$-Verteilung $E[\tilde{\mu}_1]=\mu_1$ gesichert ist. μ^*_1 ist ein gemäß (3.2.12b) aus einer vorliegenden Stichprobe errechneter (Punkt-)Schätzwert für μ_1.

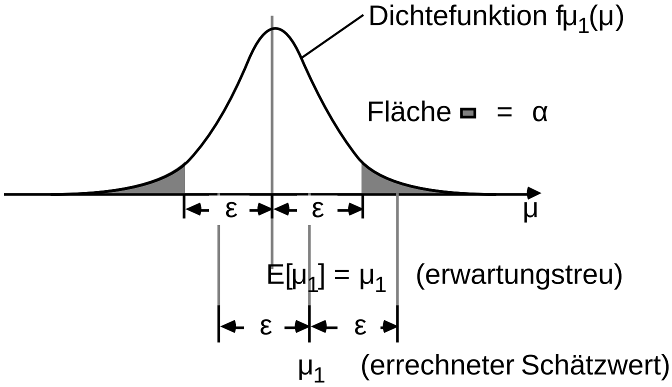


Abbildung 3.2.18: Dichtefunktion $\tilde{\mu}_1$ und Konfidenzintervall $\mu^*_1 \pm \varepsilon$

Die gestellte Frage (3.2.17) hat zwei Lesarten:

(a) Wie groß ist die Wahrscheinlichkeit, daß ein (errechnetes) μ^*_1 aus dem Intervall $(\mu_1-\varepsilon,\mu_1+\varepsilon)$ herausfällt?

(b) Wie groß ist die Wahrscheinlichkeit, daß das Intervall $(\mu^*_1-\varepsilon,\mu^*_1+\varepsilon)$ das (wahre) μ_1 nicht enthält?

Da wir ja μ_1 nicht kennen, wohl aber μ^*_1, ist Lesart (b) hilfreicher. Die Beantwortung der Frage ist allerdings aus Lesart (a) gewinnbar, die als offensichtliche Antwort liefert: "Entsprechend der Fläche unter der Dichtefunktion von $\tilde{\mu}_1$ ausserhalb dieses Intervalls" - in Abb. 3.2.18 also entsprechend der schraffierten Fläche.

Gesetzt den Fall, wir kennen diese Fläche oder eine obere Schranke α für sie, so daß (3.2.17) beantwortbar ist zu

(3.2.19) $P[\,|\tilde{\mu}_1-\mu_1|\geq\varepsilon\,]\leq\alpha$

Intervall-schätzer
Konfidenz-Intervall

Nun sind ε und α sicher voneinander abhängig. Bisher haben wir von gesetztem ε aus in Richtung des zugehörigen α argumentiert. Üblicherweise wird die Frage aber umgekehrt und von einem gesetzten α aus das zugehörige ε gesucht mit der Argumentation: Wenn ich mit (z.B.) Wahrscheinlichkeit 0.9 (für 90% aller potentiell gezogenen Stichproben) sicher gehen will, daß das wahre μ_1 um nicht mehr als ε vom errechneten Wert μ^*_1 entfernt liegt, (α=0.1), auf welches ε muß ich dann gefaßt sein? Man nennt $\mu^*_1\pm\varepsilon$ ein $(1-\alpha)\cdot 100\%$-**Konfidenzintervall** für μ_1; $(\tilde{\mu}_1-\varepsilon,\tilde{\mu}_1+\varepsilon)$ ist ein sog. **Intervallschätzer**.

Tschebyscheff-Ungleichung

Bleibt "nur noch" die Frage zu klären, welche (ε,α)-Paare denn zusammengehören. Eine diesbezügliche, sehr allgemeine Antwort liefert die sog. **Tschebyscheff'sche Ungleichung** (Sie erinnern sich!), nach der für eine beliebige ZV X mit (existierenden) Erwartungswert E[X] und Varianz V[X] gilt, daß (für beliebiges positives c)

(3.2.20) $P[\,|X-E[X]|\geq c\,]\leq V[X]/c^2$

Anwendung

Für unsere konkrete Erwartungswertschätzung erhalten wir mit (3.2.13,3.2.16):

(3.2.21) $P[\,|\tilde{\mu}_1-\mu_1|\geq\varepsilon\,]\leq V[Y]/(n\varepsilon^2)$

Das Hantieren mit dieser Ungleichung ist recht einfach:

Beispiel 3.2.22: Gesetzt den Fall, wir sind an einem 90%-Konfidenzintervall für μ_1 interessiert; also mit (3.2.19, 3.2.21) an $\alpha=V[Y]/(n\varepsilon^2)=0.1$. Gesetzt, wir hätten eine Stichprobe vom Umfang 10 vorliegen, dann ist $V[Y]/(10\cdot\varepsilon^2)=0.1$, d.h. $\varepsilon^2=V[Y]$ bzw. $\varepsilon=\sigma[Y]$. Bei errechnetem μ_1-Punktschätzwert μ^*_1 ist also $(\mu^*_1-\sigma, \mu^*_1+\sigma)$ das gesuchte Konfidenzintervall.

Testfrage 3.2.23: Welches 90%-Konfidenzintervall erhalten Sie beim Stichprobenumfang 1000? Welches 95%-Konfidenzintervall (Sie wollen "sicherer" sein) beim Stichprobenumfang 10?

Generell gesehen sollte klar sein, daß ein Intervallschätzer für die Schätzung einer Größe ungleich wertvoller ist als ein Punktschätzer (angesichts der Variabilität des Punktschätzers ist ja, genau genommen, mit einer vorliegenden Punktschätzung allein recht wenig Information vorhanden!). Für unseren speziellen μ_1-Intervallschätzer auf Basis der Tschebyscheff-Ungleichung sind allerdings noch zwei (nicht ganz so schöne) Ergänzungen nachzutragen:

Kritik

σ unbekannt

- (3.2.21) benutzt die Varianz V[Y] der beobachteten Größe zur Schätzung des Konfidenzintervalls - dieses Charakteristikum der (Modell-)ZV Y ist uns aber unbekannt! Wir könnten versucht sein, statt V[Y] den zugehörigen Schätzer $\tilde{\sigma}^2$ nach (3.2.15) zu verwenden. Damit verlassen wir aber die Voraussetzungen der Tschebyscheff-Ungleichung (3.2.20)!

Pessimismus

• Die Tschebyscheff-Ungleichung liefert, wie sich durch konkretere Untersuchungen zeigen läßt, ein recht "breites" Konfidenzintervall. Dies ist verständlich, da sie ganz allgemein gilt und keinerlei Information über die Verteilung der Beobachtungsgröße (bei uns: Y) oder die Verteilung des Schätzers (bei uns: $\tilde{\mu}_1$) verarbeitet. Umgekehrt gesagt: Bei Vorliegen und Benutzung derartiger Informationen sollten auch bessere ("schmalere") Konfidenzintervalle ermittelbar sein.

Rechtfertigung

Zusammenfassend liefert uns der "pessimistische" Charakter des Tschebyscheff-Konfidenzintervalls (wahre Konfidenzintervalle vermutlich deutlich kleiner) doch eine gewisse Berechtigung, die durch Verwendung des Schätzers $\tilde{\sigma}^2$ für σ^2 (vermutlich) "eingeschleppte" Ungenauigkeit als kaschiert anzusehen und, für jetzt, (3.2.21) mit substituiertem $\tilde{\sigma}^2$ als μ_1-Intervallschätzer zu tolerieren.

Testfrage 3.2.24: Gegeben (beobachtet) sei die folgende zufällige Stichprobe von 30 Realisierungen einer ZV Y:

Messungen y_i, i=1,2,…,30:

4.35	5.29	4.43	5.54	6.08
5.34	6.84	4.87	5.63	4.80
5.00	6.40	1.95	3.40	4.77
4.79	4.29	5.85	5.13	4.21
3.55	6.56	5.60	3.61	6.64
3.95	4.57	6.02	2.98	3.62

(i) Schätzen Sie erstes und zweites Moment von Y.

(ii) Zeigen Sie, daß die Form

$$\tilde{\sigma}^2 = (\sum Y_i^2 - n \cdot \tilde{\mu}_1^2)/(n-1)$$

identisch ist zu (3.2.15). Warum ist diese Form für praktische Zwecke geschickter? (Denken Sie an eine sequentielle Gewinnung der Stichprobe.)

(iii) Schätzen Sie die Varianz von Y.

(iv) Geben Sie ein 90%-Konfidenzintervall für den Mittelwert-(Erwartungswert-)Schätzer an.

Einschätzung Ergebnisse

"Ein wenig Statistik" heißt dieser Abschnitt. Wenn wir uns jetzt Abschn. 3.1 nochmals vornehmen (dieser wies uns ja deutlich auf die Notwendigkeit der Beschäftigung mit Statistik hin) und insbesondere dessen Ende, sehen wir, daß wir inzwischen verwertbare Kenntnisse dazugewonnen haben, aber tatsächlich nicht viele. Für beide als notwendig erkannten Einsatzgebiete (Beobachtung von Lasten und Beobachtung von Experimenten) können wir Punktschätzer für die Momente von (als eindimensionale ZV modellierten) Beobachtungsgrößen angeben; zusätzlich einen Punktschätzer für die Varianz und einen Intervallschätzer für den Erwartungswert; alles für unabhängige Beobachtungsfolgen (unabhängige Stichproben).

Anwendung auf Beispiele:

Sehen wir uns kurz die einschlägigen Beispiele aus Abschn. 3.1 an:

Last

• Die Beschreibung einer repräsentativen Last (zum Zwecke der Reproduktion dieser Last im kontrollierten Objekt-Experiment) muß von der Beobachtung realer Lasten ausgehen. Mögen dies z.B. batch-Lasten sein, bestehend aus einer

Folge real abgearbeiteter jobs. Jeder job benötigt (eine mögliche Form der Charakterisierung:) bestimmte Betriebsmittel in bestimmtem Umfang; diesbezügliche Angaben sind mittels Software-Monitor (schon in der Basisform der accounting-software) gewinnbar; so benötigt etwa jeder job eine gewisse CPU-Zeit zur Fertigstellung - nach Beobachtung eines Betriebsmittelintervalls liegt eine Folge von job-CPU-Zeiten (Stichprobe!) vor. Ist die Stichprobe unabhängig? Da wir keinen Grund haben anzunehmen, daß Benutzer bei Formulierung ihrer Programme auf Programme anderer Benutzer achten (so daß die CPU-Zeit des i-ten jobs von den jobs der Vorgänger unbeeinflußt sein sollte), können wir die Frage bejahen und sind demnach in der Lage, mit den Resultaten dieses Abschnitts den mittleren CPU-Bedarf **des** typischen jobs dieses Betriebsmittelintervalls zu schätzen und ein Konfidenzintervall für den Mittelwert anzugeben. Unsere Wünsche der Charakterisierung von Betriebsmittelbedürfnissen von jobs könnten natürlich anspruchsvoller sein - wir kommen im nächsten Abschnitt darauf zu sprechen.

Leistung

• Die Beurteilung eines Objekt-Experiments fußt auf Beobachtung von Leistungskriterien. Wir hatten als Beispiel wiederholt auf job-Verweilzeiten abgehoben. Nach Ablauf des Experiments liegt uns (per Software-Monitor erhoben) dafür eine Folge von job-Verweilzeiten (Stichprobe!) vor. Ist die Stichprobe unabhängig? Höchstwahrscheinlich nicht, wie wir uns leicht überlegen können: Zwar sind die Charakteristika des jobs selbst (oben hatten wir über seinen CPU-Bedarf geredet) von denen anderer jobs als unabhängig anzusehen. Diese Charakteristika beeinflussen seine Verweilzeit. Darüber hinaus ist seine Verweilzeit aber von seiner Last-Umgebung abhängig; und hier entsteht Abhängigkeit! Hatte nämlich ein job eine lange Verweilzeit, und war er aufgrund großer eigener Bedürfnisse selbst "schuld" daran, dann werden seine Nachfolger höchstwahrscheinlich auch längere Verweilzeiten aufweisen, weil sie (auch wenn ihre Bedürfnisse gering waren) in dem (durch den hohen Bedarf "verstopften") System lange warten mußten. Dies als eine (von einer Reihe ähnlicher) der Argumentationen dafür, daß Folgen von Verweilzeiten i.allg. **positiv korreliert** sind: Auf große folgen oft wieder große, auf kleine oft wieder kleine Verweilzeiten. Wir können unsere Resultate also nicht ohne weiteres einsetzen, um z.B. ein Konfidenzintervall für **den** Verweilzeitmittelwert auszurechnen. Eine Abhilfe: Wir dürfen erwarten, daß job i einen Einfluß auf die Verweilzeit von job i+1 hat (Numerierung in der Reihenfolge der Fertigstellung); auch einen auf job i+2, aber wahrscheinlich einen geringeren; einen weiter fallenden auf job i+k, mit steigendem k. Bei genügend großem k sollte der sequentielle Einfluß ausgestorben sein. Wenn wir also (eine neue Art des "sampling") nicht jede job-Verweilzeit in unsere Stichprobe aufnehmen, sondern jede dritte (oder fünfte, oder zehnte?), sollten wir schließlich eine unabhängige Stichprobe vorliegen haben. So viel zur Idee, deren Ausfüllung sicher einer Methode bedürfte festzustellen, ob die Ahnung aussterbenden sequentiellen Einflusses auch trägt (bzw. ab welchem sampling-Abstand sie trägt).

Ausblick

Wir müßten auch in anderem Kontext auf statistische Überlegungen vertiefter eingehen und zwar bei der (später zu skizzierenden) Auswertung von Ergebnissen stochastischer Simulatoren - methodisch gesehen eine Aufgabe, die der hier behandelten Auswertung von Objekt-Experimenten sehr ähnlich ist. Auch dort allerdings werden wir sagen müssen: "ein wenig" Statistik; und auf die vielen existierenden Statistik-Bücher verweisen.

3.3 Last-Modellierung

Last-Reproduktion

Es ist in den vergangenen Abschnitten wiederholt angeklungen, und Ihnen daher sicher inzwischen deutlich geworden, daß die Aufbringung einer Last ein wesentliches Problem bei der Durchführung von Objekt-Experimenten darstellt. Wir hatten uns schon darüber verständigt, daß wir es selbst bei Konzentration auf eine konkrete RS-Installation nicht mit "einer Last" zu tun haben, sondern mit einer Reihe durchaus verschiedener Last-Situationen. Die potentielle Vielfalt von Last-Situationen ist sicher für einen RS-Hersteller / -Vertreiber (verschiedene Benutzungsumgebungen, sprich: Kunden) noch deutlich größer. Wir hatten ferner erkannt, daß im Hinblick auf vergleichbare bzw. reproduzierbare Objekt-Experimente (veränderliche Maschine) die Darstellung der "gleichen", d.h. einer reproduzierbaren Last fast unabdingbar ist, was uns von den Reallast-Experimenten zu Experimenten mit gesetzter, kontrollierter Last führte. Es ist auch unmittelbar verständlich, obzwar bisher nur implizit erwähnt, daß wir bei dem Versuch, eine für den Realbetrieb typische Last-Situation zu Experimentzwecken zu reproduzieren, zwei je für sich anspruchsvolle Aufgaben zu bearbeiten haben, nämlich

Probleme:

(i) die Beobachtung (Messung) des Realbetriebs, nachfolgende Analyse der beobachteten Last und letztliche **Charakterisierung von** (für bestimmte Last-Situationen) **repräsentativen Lasten**; *Charakterisierung*

(ii) die **Darstellung** / Implementierung lauffähiger Lasten auf Basis ihrer repräsentativen Charakterisierungen und letztliche **Aufbringung dieser reproduzierbaren Lasten** im Experiment (Exklusiv-Betrieb!). *Darstellung und Aufbringung*

Bei dieser Formulierung der Aufgaben hat uns, im Vergleich zu unserem Überblick über leistungsorientierte Problemstellungen (Kap. 1), eine der möglichen Problemstellungen stark die Hand geführt, nämlich die Frage nach dem Leistungsverhalten unterschiedlicher Maschinen(-Versionen) unter der gleichen Last. Diese Frage ist typisch für Auswahlsituationen (Kauf, Anmietung von RS) sowie für Verbesserungssituationen (tuning, upgrading von RS). Machen wir uns kurz klar, daß auch die verbleibenden Problemstellungen, nämlich Fragen nach dem Leistungsverhalten einer festen Maschine unter unterschiedlichen Lasten (bzw. von mehreren verschiedenen Maschinen unter mehreren unterschiedlichen Lasten) die nämlichen Aufgaben aufwerfen (und sogar noch zusätzliche): Die "verschiedenen Lasten" werden entweder repräsentativ für Real-Lasten sein (also typisch für beobachtbare Last-Situationen aus einer oder mehreren RS-Installationen): Beobachtung, Charakterisierung, Darstellung und Aufbringung der Lasten ist für Objekt-Experimente nötig. Oder aber die verschiedenen Lasten werden auf Ahnungen / Befürchtungen / Ideen beruhen, wie es in Prognose-Situationen normal ist (prognostizierte Evolution der Last und Überprüfung des erwarteten Leistungsverhaltens in - z.B. - 12 Monaten, prognostizierte Last aufgrund zusätzlicher - neuer - DV-Aufträge und Überprüfung des Leistungsverhaltens): Beobachtung, Charakterisierung, jetzt zusätzlich: Veränderung der Charakterisierung auf Prognose-Basis, Darstellung und Aufbringung der Lasten ist zur Durchführung von Objekt-Experimenten erforderlich.

Last-Darstellung

Stützen wir uns für unsere weiteren Überlegungen zunächst auf die zweite der gelisteten Aufgaben, die Darstellung und Aufbringung von Lasten. Wir erkennen schnell, daß die Darstellung einer (lauffähigen!) Last allemal aus einem oder mehreren Programmen (auf irgendeiner sprachlichen Ebene) bestehen muß: Wie sonst könnte sie "lauffähig" sein? Wir erkennen ebenso schnell, daß wir es bei der Last-Darstellung durch Programme (praktisch gesehen:) umso schwieriger haben, je "echter", "realer" diese Programme sind; denn: Ein echtes Programm hin- *reale Programme?*

terläßt i.allg. (beabsichtigterweise!) seine Spuren in Form geänderter Dateien (Daten- und Programmbestände); wir werden es nicht einfach, zu Experimentzwecken, einige Male starten können, ohne Ärger mit den betroffenen Benutzern zu bekommen! Weiter: Unter den echten Programmen sind u.U. solche, die (insbesondere im Hinblick auf die Zugriffsberechtigung auf Datenbestände) einer bestimmten Vertraulichkeit/Geheimhaltung unterliegen; diese werden wir nicht einfach, zum Experimentieren, in die Hand bekommen! Schließlich: Unter den beobachteten echten Programmen sind evtl. auch solche, die wir für "untypisch" halten und eliminieren wollen; es sind i.allg. viele Programme, und wir wollen im Experiment vielleicht mit wenigen arbeiten, u.s.f.

prototypische Programme: Last-Modelle

Worauf ich hinaus will: Obwohl wir darauf aus sind, repräsentative Lasten zu verwenden, werden diese zwar (notwendig!) in irgendeinem Sinne sehr ähnlich sein zu (beobachteten oder prognostizierten) realen Lasten, aber (aus praktischen Gründen!) keine oder wenige echte Programme enthalten. Die für Experimente benutzten lauffähigen Last-Darstellungen werden allemal aus Programmen bestehen, aber eben oft nur "prototypischen", in "prototypischer" Zusammenstellung: Wir arbeiten mit Darstellungen, die unsere charakterisierten Lasten nur repräsentieren, eben mit (vgl. Titel dieses Abschnitts) **Last-Modellen.** Neben gewissen lauffähigen Programmen müssen Last-Modelle eine weitere wichtige Komponente enthalten, um eine Last darzustellen: Es muß klar sein, wann (bzw. unter welchen Umständen) welches der Modellprogramme zu starten/zu initiieren ist. Mehr dazu nach einigen Worten über die Aufbringung der Last.

Last-Aufbringung

Aufbringung der Last: Initiierung definierter Programme zu definierten Zeitpunkten. Im Realbetrieb ist genau dies die Rolle der RS-Benutzer. Im Objekt-Experiment muß diese Aufgabe simuliert werden. Eine vielleicht erste Idee, dazu die Operateure des RS zu verwenden (geplantes manuelles Starten der Programme von System-Konsolen aus) und zusätzlich menschliche Helfer zur Abdeckung der interaktiven Belastung einzusetzen (geplante Manipulationen von terminals aus) sollte man angesichts der geforderten Reproduzierbarkeit des Experiments (d.h. der Last) schnell wieder vergessen: Zu irrtumsanfällig und zu unpräzise (Zeitpunkte!) würde der simulierte Betrieb dadurch. Ausweg kann nur sein, die Aufbringung der Last zu automatisieren. Was heißt das? Nun, die Fähigkeiten von RS dazu auszunutzen, um Programme zu präzisen Zeitpunkten/unter präzise definierten Umständen zu starten. Realiter kann dies, je nach RS (samt seiner System-Software!), auf diverse Arten geschehen. Breit einsetzbar sind dazu Programme in der Kommandosprache des RS (in dieser kann man ja den Start von Programmen in Form von Kommandos formulieren), die allerdings notwendigerweise Kommandos zur Kontrolle der Zeit enthalten muß: "Warte für (z.B.) 2 min", "Warte bis (z.B.) 15.05 Uhr" und ähnliche Vorschriften müssen als Kommando formulierbar sein, um etwa der (Last-Modell-) Anweisung "Programm x starte (z.B.) 2 min nach Programm y" Folge leisten zu können. Man nennt diese (den zeitlichen Ablauf der Last-Aufbringung kontrollierenden) Programme auch **Treiber**(-Programme). Treiber von zweierlei Art sind im Gebrauch: interne und externe. Interne Treiber "laufen" auf dem (zu beobachtenden) Objektsystem; offensichtlicher Punkt zur Warnung: Damit bedienen sie sich auch der Ressourcen des Objektsystems und verfälschen in gewissem Umfang den Betrieb (Analogon: SW-Monitore, Kap. 2). Externe Treiber laufen auf einem "anderen" RS (einem zweiten autonomen Rechner, einem etwa ohnehin vorhandenen Vorrechner, einem intelligenten terminal, ...), das mit dem Objektsystem natürlich derart kommunizieren kann, daß es Startanweisungen für Programme zu senden (und u.U. Antworten zu empfangen) in der Lage ist.

Nutzung des Rechners · *Treiber-Programme*

Last-Ebenen

Denken wir noch etwas über Darstellung und Aufbringung von Lasten nach, und versuchen wir, Erkenntnisse aus dem Einführungskapitel 1 mit einzubringen: Abb. 1.13,1.14,1.16 samt Diskussion machten deutlich, daß "Last" bezüglich jeder Ebene unseres hierarchisch strukturierten RS definierbar ist. Wenn wir hier über die Darstellung von Last in Form lauffähiger Programme reden, können dies Programme verschiedener Art sein - jedes aber bezieht sich (auch sprachlich!) auf eine ganz bestimmte Ebene. Das Starten von Programmen seitens eines Treibers (in der Sprechweise des Kap. 1: der Aufruf von Diensten) erfolgt demnach zumindest gedanklich ebenfalls an einer bestimmten Ebene des hierarchischen Gebäudes. Die Dynamik des Treibers (wir wollen ja an dieser Ebene repräsentative Last produzieren) simuliert also "eigentlich" das Verhalten der "oberen Hälfte" des gesamten dynamischen Systems (jetzt ist ein Blick auf Abb. 1.12 hilfreich), bis hinauf und unter Einschluß der Systembenutzer. Diese Überlegung ist nicht reine Gedankenspielerei, sondern sollte uns auf einen Gefahrenpunkt hinweisen, der in der einschlägigen Literatur als closed-loop-effect/**Rückkopplungseffekt** bezeichnet wird. Er wird am Beispiel (Abb. 1.14/1.16) sehr deutlich: Bei der Reproduktion des Lastverhaltens bzgl. (z.B.) Ebene c muß daran gedacht werden, daß gewisse γ.-Aufrufe (Start von γ.-Programmen) nicht auf bestimmte (beobachtete) Zeitpunkte festgelegt werden dürfen; wenn sie nämlich aus β_2-Aufrufen des Musters (Programms!) α_2 resultieren, wird ihr Initiierungszeitpunkt den Bedingungen der α_2-Programmschleife gehorchen (nach Abschluß eines β_2-Dienstes wird 15 sec gewartet, dann wieder ein β_2-Dienst aufgerufen, der seinerseits γ.-Dienste aufruft ...); die Ausführungsdauer von γ.-Diensten beeinflußt also (Rückkopplung!) die Zeitpunkte des Aufrufs von γ.-Diensten. Vom "repräsentativen" Treiber muß diese Rückkopplung erfaßt werden, da er sonst (bei Experimenten mit veränderter Maschine, also z.B. einer mit verkürzter/verlängerter γ.-Ausführungszeit) keine korrekte Last produziert. Und ganz praktisch gesprochen: Insbesondere interaktive Benutzungen (und ihre Reproduktion in Experimenten) weisen dieses Rückkopplungsverhalten auf und sind daher mit der entsprechenden Aufmerksamkeit zu modellieren.

Rückkopplung

Verwendete Last-Darstellungen

Von der vorstehenden, nicht sehr konkreten, mehr aufs Prinzipielle ausgerichteten Diskussion der Aufgabe "Darstellen und Aufbringen von Lasten" jetzt ein Sprung ins andere Extrem: Zu einer Auflistung von Last-Darstellungen, wie sie in der Praxis der Objekt-Experimente üblich und gebräuchlich sind; zu recht konkreten Angaben also, die aber den Blick fürs Prinzipielle gelegentlich vermissen lassen. Aus FeSZ83 ist die Übersicht 3.3.1 über Typen von Last-Darstellungen entnommen - andere Klassifizierungs-Schemata sind durchaus denkbar. Einige Erklärungen dazu:

Real-Lasten

- **Real-Lasten** als Last-Modelle (die genaue Wiederholung der Last eines konkreten realen Betriebs-Intervalls) hatten wir aus den verschiedensten Gründen (blättern Sie zurück) als nicht empfehlenswert eingestuft.

Synthetische Lasten

- **Synthetische Lasten** entstammen im Grunde der Reallast, sind aber für experimentelle Zwecke aufbereitet/"gereinigt". Das verbreitetste Beispiel hierfür ist der sog. **benchmark**, der vor allem im Auswahlbereich (Kauf-/Konfigurierungs-Entscheidungen) im Gebrauch ist, dort i.w. auf batch-Lasten spezialisiert. Der **natürliche** benchmark besteht aus einer gewissen Menge realer batch jobs (oder auch: "job steps", bei strukturierten jobs). Die verwendeten jobs sind insofern "gereinigt", als sie keine permanenten Auswirkungen auf Dateien hinterlassen, auch werden jobs bzgl. vertraulicher Informationen nicht im benchmark enthalten sein. Die Auswahl des benchmarks aus der Reallast erfolgt "repräsentativ" - entweder indem, auf der Ebene auszuführender Funktionen, Test workloads / Last-Modelle

- Real / real
- Synthetic / synthetisch
  - Natural (benchmarks) / natürlich
  - Hybrid / hybrid
- Artificial / künstlich
  - Executable / ausführbar
    - Instruction mixes
    - Synthetic programs
    - Kernels
    - Synthetic scripts
    - Interactive drivers
    - Traces
    - …
  - Nonexecutable / nicht ausführbar
    - Analytic models
    - Distribution-driven models
    - …

Abb. 3.3.1: Klassifikation von Last-Modellen; aus FeSZ83

eine typische Zusammenstellung von (z.B.) COBOL-COMPILEs, FORTRAN-EXECUTEs, … angestrebt wird oder indem, auf der Grundlage des Betriebsmittelbedarfs, eine typische Zusammenstellung von jobs verschiedener Betriebsmittelanforderungen (CPU-Zeit, Zahl E / A-Operationen, Arbeitsspeicherumfang, …) erfolgt. Es dürfte klar sein, daß (auch für alles Folgende) eine Charakterisierung auf Grundlage von Betriebsmittelanforderungen (eine **physikalische** Charakterisierung) dort versagt bzw. in Schwierigkeiten gerät, wo im Verlauf der Experimente Hardware-Änderungen untersucht werden (z.B. Austausch CPU), während Charakterisierungen auf der Basis auszuführender Funktionen (**funktionale**, auch: **logische** Charakterisierungen) ein breiteres Potential an Repräsentativität aufweisen. Zu den ausgewählten jobs des benchmarks ist (wie wir wissen) als zusätzliche Information der Zeitpunkt ihres Starts (im Experiment) festzulegen. Hier macht man es sich oft einfach, indem man (bei benchmarks geringen Umfangs) alle jobs "zugleich" startet (als absolut minimale Festlegung ist aber die Reihenfolge der jobs über alle Experimente beizubehalten). Das ist offensichtlich kein (experimenteller) Betrieb, der unserer Vorstellung einer "gleichmäßigen" Last-Situation entspricht; das zu beobachtende Leistungs-Kriterium ist dann auch oft beschränkt auf "Gesamtabwicklungszeit des benchmarks" - unsere statistischen Überlegungen werden weithin überflüssig (die Aussage der Experimente ist aber auch dementsprechend "schmal").

Testfrage 3.3.2: Wie können Sie im Rahmen der Verwendung eines benchmarks (= ausgewählte Menge realer jobs) einen experimentellen Betrieb aufbauen, der eine "regelmäßige, gleichmäßige" Belastung für ein gewisses (Experiment-) Betriebsintervall zumindest annäherungsweise aufrecht erhält?

Das Analogon zum natürlichen benchmark im batch-Bereich ist auf der Seite interaktiver RS-Benutzung (Rückkopplung zu beachten!) das natürliche **Skript:** Idee ist hier, den real stattfindenden Ablauf von terminal-"Sitzungen" (sessions) interaktiv arbeitender Benutzer aufzuzeichnen (und im Experiment zu reproduzieren). Die Darstellung eines Skripts enthält also die präzise Folge eingegebener Kommandos **und** die "Denk"-Zeiten, die zwischen Antworten des Systems und darauf folgendem Benutzerkommando verstreichen. Die Idee der "gleichmäßigen" Last-Situation ist hier (naheliegend!) verfolgbar durch Ingangsetzen (Treiber!) mehrerer unterschiedlicher (typischer!) Skripts und ihrer laufenden Wiederholung.

Die Bezeichnung **hybride** synthetische Lasten soll die potentielle Verwendung sowohl natürlicher als auch (im nächsten Punkt zu besprechender) künstlicher Last-Komponenten im Rahmen eines Last-Modells andeuten.

Künstliche Lasten

• **Künstliche Lasten** sind Last-Modelle in dem Sinne, daß ihre Programme (im Falle **ausführbarer** Lasten) nicht der Real-Last entnommen, sondern eigens zur (repräsentativen!) Darstellung von Experimental-Lasten erstellt werden. Daß hier auch **nicht ausführbare** Last-Darstellungen auftauchen, verstehen Sie bitte als Vorgriff auf die folgenden Kapitel, für die wir uns (vgl. Kap. 1) das Thema Modell-Experimente (statt der hier besprochenen Objekt-Experimente) vorgenommen haben; auch Modell-Experimente werden Last-Darstellungen benötigen - nur werden sie im Unterschied zu hier nicht notwendig ausführbar sein müssen, nicht notwendig aus Programmen zu bestehen haben. Die ausführbaren künstlichen Last-Programme sind (wie alle Programme) auf eine bestimmte Ebene des hierarchischen RS bezogen. Beim **Instruktions-Mix** ist dies die Ebene der Maschinensprache, die Ebene der direkt von einer CPU interpretierbaren Programme. Genau genommen ist ein Mix gar kein Programm, sondern eine Auflistung der relativen Häufigkeiten, mit denen die diversen Befehle / Instruktionen einer Maschinensprache in der zu modellierenden Realität auftreten. Diese Häufigkeiten werden je nach Benutzerumgebung unterschiedlich sein (korrekterweise: bestimmte Last-Situationen!). Als (nicht unrealistisches) Beispiel für eine technisch-wissenschaftliche Benutzerumgebung: 4% Gleitpunkt-Additionen / Subtraktionen, 8% Gleitpunktmultiplikationen, ..., 18% Transporte zwischen Register und Arbeitsspeicher (beide Richtungen), 20% bedingte Sprünge, ... Der historisch bekannteste derartige Mix ist der sog. Gibson-Mix, Gibs70; er ist allerdings wegen seines "Alters" für heutige Rechnerarchitekturen und heutige Belastungen nicht mehr einsetzbar.

instruction mix

Ein Instruktions-Mix kann nun sicher dazu benutzt werden um, am Schreibtisch, aus den Hardware-Spezifikationen einer CPU (Zahl von Takten für bestimmte Befehle plus Taktzeit) eine MIPS- / MOPS- / MFLOPS-Zahl (vgl. Kap. 1) auszurechnen. Um jetzt aus einem Mix ein ausführbares Programm (für Objekt-Experimente) zu konstruieren, müssen ("künstliche") Befehlsfolgen notiert werden, in denen bei Ausführung die diversen Instruktionen mit spezifizierter Häufigkeit aufgerufen werden. Damit hat man den Schritt zum **synthetischen Programm** vollzogen. Synthetische Programme enthalten häufig einige ineinandergeschachtelte Schleifen; die Zahl der Schleifendurchläufe auf verschiedenen Schachtelungstiefen ist meist über Parameter einstellbar, so daß sich die relativen Häufigkeiten des Aufrufs diverser Instruktionsarten (in Grenzen) verändern und einer etwa gemessenen Mix-Information anpassen lassen. Im Hinblick auf die heute übliche, nahezu ausschließliche Verwendung höherer Programmiersprachen liegt es nahe, die Idee des Instruktions-Mix auf der Ebene höherer Sprachen zu wiederholen, also relative Häufigkeiten der Verwendung bestimmter (höherer) Sprachkonstrukte festzustellen und im Experiment zu reproduzieren. Wir sprechen jetzt (im Gegensatz zum Instruktions-Mix) von einem **Statement-Mix**. Ein zugehöriges synthetisches Programm (das diesen Mix zu reproduzieren hat) ist auf der Ebene einer höheren Programmiersprache notiert; das Objekt-Experiment unter Einsatz dieses Programms stellt somit (nachdenken!) sowohl die Qualität des verwendeten Compilers fest (wie "effizient" ist der erzeugte Objekt-Code?) als auch die Effektivität der CPU (wieviel Objekt-Code-Instruktionen werden pro Zeiteinheit ausgeführt). Bekannte synthetische Programme auf Basis von gemessenen statement-Häufigkeiten sind der sog. Whetstone Benchmark, CuWi76 (bzgl. mathematisch-technischer Benutzerumgebungen) und der sog. Dhrystone Benchmark, Weic84 (bzgl. System-Software ausgelegt).

statement mix

kernel

Etwa dem gleichen Zweck wie die besprochenen (z.B. auf gemessenen Mix-Zahlen beruhenden) synthetischen Programme dienen die sog. **kernel**(s). Bei ihnen handelt es sich um Ausschnitte aus realen Programmen, Ausschnitte, die wegen ihrer häufigen Verwendung als typisch für die Belastung angesehen werden; als mögliche Beispiele: Sortierprogramme, Matrix-Multiplikationen, sequentielles Datei-Lesen, … Die Ausschnitte sind allerdings so verändert, daß sie i.allg. keine irgendwie sinnvollen Resultate liefern, sondern eben nur typische Benutzungsmuster festhalten; sie werden darüber hinaus oft parametrisiert, um die "Größe" der von ihnen behandelten Probleme einstellen zu können: Anzahl der zu sortierenden Größen, Dimension der Matrizen, Länge der Datei, … Die künstliche Last besteht nun aus verschiedenen solcher kernels, die wiederholt (und mit "typisch" variierenden Problemgrößen) gestartet werden. Beispiele für real verwendete kernels im mathematisch-wissenschaftlichen Bereich sind der Linpack benchmark, Dong88, und die Livermore kernels, MMah88; speziell im Datenbankbereich sei auf die TPC-Initiative (Anon85, TPC94) aufmerksam gemacht, die regelmäßig Daten über gemessene Werte das Effizienz-Maßes "transactions per second (tps)" bzgl. konkreter DB-Systeme und konkreter Rechnerinstallationen berichtet.

script

Das Gegenstück zu den künstlichen, synthetischen Programmen (zur Darstellung von batch-Lasten) sind auf der Seite interaktiver Belastung die **synthetischen Skripts**. Waren bei natürlichen Skripts reale Kommandofolgen (samt Denkzeiten) reproduziert worden, so stellt man hier künstliche Kommandofolgen auf mit dem Bestreben, die relativen Kommandohäufigkeiten der Realität zu wiederholen. Grundlage sind beobachtete "Mixes" auf Kommandoebene (und beobachtete Zwischen-Kommando-Denkzeiten).

Warnungen

Im Hinblick auf "öffentlich gehandelte" Vergleichs-(Leistungs-)Zahlen wie etwa die diversen MIPSe ist große Vorsicht geboten, was ihre (oft unerwähnte) Definition anbetrifft. Ich möchte dies an drei besonders augenfälligen Details klar machen:

- Es ist offensichtlich fragwürdig, im Vergleich von RISC- und CISC-Architekturen reine MIPS-Zahlen (nochmals: Mega Instructions Per Second) zu verwenden: RISC-Instruktionen sind "einfach", dauern (so entworfen!) eher "kurze" Zeit und erlauben folglich leicht die Angabe einer "höheren" MIPS-Zahl; dafür benötigt aber ein Programm zur Bewältigung einer bestimmten festen Aufgabe die Ausführung einer größeren Anzahl von RISC-Instruktionen. Welche CISC-MIPS-Zahl welcher RISC-MIPS-Zahl (bezogen auf, z.B., die Ebene einer verwendeten HLL) entspricht, ist im Prinzip nicht generell zu sagen, sondern hängt zum einen von der Verwendungshäufigkeit der diversen HLL-Strukturen ab (also, wenn man nicht präziser werden möchte, vom "Statement-Mix"), zum anderen aber auch stark vom eingesetzten Compiler.

• Heutige Rechnerarchitekturen setzen in steigendem Maße "Caches" (schnelle Pufferspeicher zwischen Arbeitsspeicher und CPU) ein. Ziel dieser Maßnahme ist es, die (typischerweise vorliegende) Instruktions- und Daten-Lokalität von Programmen zu nutzen und die Programme (nach möglichst nur einmaligem "Laden") weitgehend "aus dem Cache heraus" auszuführen. Da der überwiegende Anteil der Ausführungszeit einer CPU-Instruktion (d.h. der Dauer des Instruktionszyklus) aus fetch / store-Operationen (auf Register-Transfer-Ebene) besteht, ist mit dieser Maßnahme u.U. eine drastische Verkürzung der Instruktionsausführungszeiten verbunden, d.h. eine drastische Erhöhung von MIPS-Zahlen. Dies ist uns zwar durchaus willkommen, beim Vergleich verschiedener Rechner aber wird damit die Größe des Caches zur (fast) allein ausschlaggebenden Größe. Und weiter: "Irgendein" Programm, auch wenn es entsprechend eines repräsentativen Statement-Mixes aufgebaut ist, taugt nicht automatisch als Last-Modell für Vergleiche! Das repräsentative Last-Modell muß offensichtlich auch eine repräsentative Lokalität aufweisen, um den Cache-Einfluß repräsentativ zu testen. Noch weiter: Auch bei Einsatz von Benchmark- und Kernel-Programmen muß darauf geachtet werden, ob sie (konstruiert als eher einfacher Code) nicht eine zu hohe Lokalität besitzen (Nachdenken!).

• Wenn man denn nun offensichtlich genötigt ist, bei moderneren Architekturen ausgefeilte Benchmark- und Kernel-Programme zur Basis des Vergleichs ihrer Leistungsfähigkeit zu machen, liegt es nahe, weltweit möglichst dieselben Programme zu verwenden, mit anderen Worten, Benchmarks und Kernels zu standardisieren. Dieser Vorgang ist de facto in voller Entwicklung (s. etwa die Übersichten Weic91, Weic90, welche auch Information über die interessante SPEC-Benchmark-Initiative enthalten). Nur: Mit solcher Standardisierung ist auch die Versuchung für Rechner- (und Compiler-) Hersteller groß, optimierende Compiler zu konstruieren, die gerade die Besonderheiten eines solchen Standard-Benchmarks in eine hohe MIPS-Zahl umsetzen (und welche "allgemeine" Software "vielleicht" weniger effizient berücksichtigen).

Soviel als knappe Übersicht. Genaueres und Ausführliches entnehmen Sie bitte (sofern Sie sich dafür interessieren, d.h. spätestens wenn Sie diesbezüglich beruflich gefordert werden) der Literatur (so: Ferr78, FeSZ83, Weic91, ACM92, DoGe93).

Charakterisierung: Verweis

Einen ähnlichen Literaturhinweis muß ich hier aus Platzgründen hinsichtlich der zweiten wesentlichen Aufgabe der Last-Modellierung (vgl. Anfang dieses Abschnitts) anbringen, der Charakterisierung realer Lasten (als notwendige Grundlage für Darstellung und Aufbringung experimenteller Lasten). Hier kommen Methoden der angewandten Statistik in großer Vielfalt zum Tragen, gilt es doch eine "repräsentative" Charakterisierung des Benutzerverhaltens zu gewinnen, das doch nicht das Benutzerverhalten selbst (also die realen Programme) sein soll. Repräsentativität im statistischen Sinne ist zu fordern, statistische Charakterisierungen von Lasten sind aus der Beobachtung realer Lasten zu identifizieren und die zugehörigen Parameter zu schätzen. Dieses Thema füllt leicht einen gesamte Vorlesung! Wie angedroht, ich verweise auf die einschlägige Literatur, insbesondere FeSZ83 (mit mehreren -zig Weiter-Verweisen).

Ich wiederhole hier auch meinen Hinweis auf Dirl94 (Literatur Kap.1). Dieses Buch beschreibt auf Basis der DIN-Norm DIN 66273 eine spezielle, standardisierte Form von Last-Beschreibungen (-"Modellen"), die Definition von Leistungsmaßen sowie die Durchführung und Auswertung von Messungen ("Objektexperimenten").

Literatur zu Kap. 3

ACM92 ACM; Reports on benchmarks and benchmark results;
Performance Evaluation Rev. vol.19(1992) nr.3 pp.36-44

Anon85 Anon. et al; A measure of transaction power;
Datamation vol.31.1(1985) nr.7 pp.112-118

CuWi76 Curnow,H.J. / Wichmann,B.A.: A synthetic benchmark;
Computer Journal vol.19(1976) pp.43-49

Dong88 Dongarra,J.J.; The LINPACK benchmark: An explanation;
in: vanderSteen,A.J.; Evaluating supercomputers; Chapman and Hall 1990

DoGe93 Dongarra,J.J. / Gentzsch,W.; Computer benchmarks; Elsevier 1993

Dörf78 Dörfler,W.: Mathematik für Informatiker; Band 2, Methoden aus der Analysis;
Hanser 1978 (erste! Auflage)

Ferr78 Ferrari,D.: Computer systems performance evaluation; Prentice-Hall 1978

FeSZ83 Ferrari,D. / Serazzi,G. / Zeigner,A.: Measurement and tuning of computer systems;
Prentice-Hall 1983

Gibs70 Gibson,J.C.: The Gibson mix; IBM TR 002043 1970

Gray92 Gray,J. (ed); The benchmark handbook for database and transaction processing systems;
Morgan Kaufman revised printing 1992

HaEK82 Hartung,J. / Elpelt,B. / Klösener,K-H.: Statistik; Oldenbourg 1982

MMah88 McMahon,F.H.;
The Livermore FORTRAN kernels test of the numerical performance range;
in: Martin,J.L. (ed); Performance evaluation of supercomputers; Elsevier 1988

MoGr63 Mood,A.M. / Graybill,F.A.: Introduction to the theory of statistics;
McGraw-Hill 1963 (bzw. Neuauflage)

TPC94 TPC; Summary of TPC results 15.3.94;
Performance Evaluation Rev. vol.21(1994) nr.3+4 pp.9-21

Weic84 Weicker,R.P.: Dhrystone: A synthetic systems programming benchmark;
CACM vol.27(1984) pp.1013-1029

Weic90 Weicker,R.: An overview of common benchmarks;
IEEE Computer vol.23(1990) nr.12 pp.65-75

Weic91 Weicker,R.: Benchmarking: Status, Kritik und Aussichten;
in: Lehmann,A. / Lehmann,F. (eds):
Messung, Modellierung und Bewertung von Rechensystemen;
IFB vol.286 Springer 1991

Leerseite

Lösungshinweise zu den Testfragen

3.1.1: Unmittelbar beobachtbar sind am ehesten die Zeitpunkte der Initiierung und der Beendigung aller jobs; eventing ist ein hoffnungsvoller Meß-Ansatz. Sie denken sicher auch sofort an einen Software-Monitor zur Messung, da es ja spezielle BS-Routinen gibt, welche die Initiierung von jobs und deren Terminierung bewerkstelligen. In diesen Routinen eingebrachte Meßstellen (software-events) können die benötigte Information (job-Identifikation, Anfang/Ende, Zeitpunkt) erfassen. Zusammengehörige (job-Initiierungs, job-Terminierungs)-Messungen müssen allerdings anschließend aus der nur zeitlich geordneten Folge von unmittelbaren Beobachtungen herausgesucht werden, um die job-Verweilzeiten (Endzeitpunkt - Anfangszeitpunkt) zu bestimmen. Dabei werden (festes Beobachtungsintervall) einige Messungen keinen Partner finden (jobs, die vor Beginn der Messung starteten und während der Messung terminierten; jobs, die während der Messung starteten und erst nach der Messung terminierten), werden also einige Meßdaten zu eliminieren sein. ***3.1.1***

3.1.3: Es gibt verschiedene Möglichkeiten, jobs derart in Klassen einzuteilen, daß die Verweilzeiten innerhalb einer Klasse vergleichbarer werden. Eine häufig benutzte Klassifizierungsmethode orientiert sich an den Betriebsmittel-Ansprüchen der jobs: Benötigte Prozessorzeit, Umfang des E/A-Verkehrs, Speicherbedarf, Druckvolumen liefern hinreichende Klassifizierungsmerkmale (z.B. Klasse 1 jobs sind jene, die nicht mehr als ... CPU-Zeit, nicht mehr als ... E/A-Transporte, ... aufweisen; Klasse 2 jobs: CPU-Zeit im Intervall ..., E/A-Transportzahl im Intervall ..., u.s.f.). Eine andere benutzte Klassifizierungsmethode ist problemorientiert: Klasse 1 sind (z.B.) PASCAL-Übersetzungen, Klasse 2 FORTRAN-Compile-Load-and-Go's, u.s.f. Für jede Klasse k würden Sie eine diesbezügliche Klassen-Verweilzeit-ZV V_k einführen, Ihre Meßergebnisse in eine Menge je klassenspezifischer Verweilzeitfolgen sortieren, um anschließend klassenspezifische Auswertungen anzustreben. Der Gesamtansatz bleibt gleich: Ein job, welcher Klasse auch immer, hat die "unsichere" Last-Umgebung; ein job, innerhalb jeder Klasse, variiert in seinen Eigenschaften - weniger zwar im Rahmen der Klassencharakterisierung, aber doch hinreichend stark, um auf Statistik nicht verzichten zu können. ***3.1.3***

3.1.4: Wir machen sofort Anleihe bei Kap. 2, wo uns die busy periods bereits ausführlich beschäftigt hatten: Die Aufzeichnung der benötigten Information $(b_1,b_2,\ldots,b_n)$, mit $b_i \triangleq$ Dauer i-tes Tätigkeitsintervall, ist uns schon in (2.1.10) begegnet. Diese Folge wäre gewinnbar aus den unmittelbar beobachtbaren und aufzuzeichnenden Ereignissen Beginn-/Ende-Zeitpunkte busy periods. Diese wiederum müßten mittels eventing-Messung gewonnen werden über Einsatz ***3.1.4***

- eines Hardware-Monitors (Zeitpunkte "Umklappen CPU-busy-bit", falls vorhanden und meßbar);
- eines Software-Monitors, der über eine Meßstelle innerhalb der Dispatching-Routine des Betriebssystems verfügt (Dispatcher: jene Komponente des BS, welche die CPU-Zuteilung(en) verwaltet).

Die busy periods variieren in ihrer Länge. Mit der Modellvorstellung, für eine Lastsituation seien bestimmte solche Längen in bestimmter Häufigkeit charakteristisch, können wir die Länge **der** typischen busy period durch eine ZV B erfassen. Obige Beobachtungsfolge $(b_1,b_2,\ldots,b_n)$ ist analog (3.1.2) zu interpretieren.

Der ebenfalls in Kap. 2 schon eingeführte sampling-Meßansatz (Beobachtung CPU busy-bit zu festen Zeitpunkten, vgl. Abb. 2.1.3; Messung per Hardware-Monitor) gestattete ja nur bei hoher Meßfrequenz, die Längen von busy periods ungefähr festzustellen. Eine direkte Interpretation der sampling-Folge sähe nun folgendermaßen aus: Eine Beobachtung des CPU busy bits zu beliebigem Zeitpunkt liefert variierende Werte b' {0,1}. Die Häufigkeiten der Werte 0 und 1 sind charakteristisch für die beobachtete Last-Situation; wir modellieren die Beobachtung durch eine ZV B', von der wir per sampling (z.B.) n Realisierungen $(b_1', b_2', \ldots, b_n')$ erfassen. Die Messungen liefern Aufschluß über den Belegt-Zustand der CPU zu beliebigem Zeitpunkt - ein Aufschluß über die Auslastung der CPU sollte sich aus den Messungen gewinnen lassen.

3.2.6 **3.2.6:**

$$\begin{aligned} V[Y] &:= E\left[\left(Y - E[Y]\right)^2\right] \\ &= E\left[Y^2 - 2\,Y\,E[Y] + E^2[Y]\right] \\ &= E\left[Y^2 - 2\,E[Y]\,E[Y] + E^2[Y]\right] \\ &= \mu_2 - \mu_1^2 \end{aligned}$$

3.2.8 **3.2.8:** Szenario 1: Der 10-te Würfel "weiß" nichts von den anderen 9 Würfeln, sein Ergebnis ist von deren Ergebnissen unbeeinflußt. Szenario 2: Der Würfel "weiß" beim 10-ten Werfen nichts von den vorangegangenen Würfen, ist also von den Vor-Ergebnissen unbeeinflußt. Die Stichprobe ist in beiden Fällen "zufällig". Für das 10-te Ergebnis sollten Sie also nichts anderes erwarten, als vom isolierten Werfen eines idealen Würfels, daß nämlich jede der Zahlen 1 bis 6 mit Wahrscheinlichkeit 1 / 6 zu erwarten ist.

3.2.14 **3.2.14:**

$$\begin{aligned} \tilde{\mu}_k &:= \frac{1}{n} \sum Y_i^k \\ E[\tilde{\mu}_k] &= \frac{1}{n} \sum E\left[Y_i^k\right] \\ &= \frac{1}{n} \sum E\left[Y^k\right] \\ &= \frac{1}{n}\; n\; E\left[Y^k\right] \\ &= \mu_k \end{aligned}$$

3.2.23 **3.2.23:**

(i)

$$\begin{aligned} \frac{V[Y]}{1000\,\varepsilon^2} &= 0.1 \\ \frac{V[Y]}{100} &= \varepsilon^2 \\ \varepsilon &= \frac{\sigma}{10} \end{aligned}$$

Konfidenzintervall: $\mu_1^* \pm \sigma / 10$

(ii)

$$\begin{aligned} \frac{V[Y]}{10\,\varepsilon^2} &= 0.05 \\ \frac{V[Y]}{0.5} &= \varepsilon^2 \\ \varepsilon &= \sigma \sqrt{2} \end{aligned}$$

Konfidenzintervall: $\mu_1^* \pm \sigma \cdot \sqrt{2}$

3.2.24: *3.2.24*

(i) Wir setzen (3.2.12,3.2.14) ein.
Mit
$\Sigma y_i = 146.06 \qquad \Sigma y_i^2 = 750.46$
und n=30 erhalten wir
$\mu_1^* = 146.06 / 30 \qquad \mu_2^* = 750.46 / 30$
$= 4.87 \qquad\qquad = 25.02$

(ii) Aus (3.2.15)

$$\tilde{\sigma}^2 = \sum \frac{\left(Y_i - \tilde{\mu}_1\right)^2}{n - 1}$$

erhalten wir

$$\tilde{\sigma}^2 = \frac{\sum Y_i^2 - 2\tilde{\mu}_1 \sum Y_i + \sum \tilde{\mu}_1^2}{n - 1}$$
$$= \frac{\sum Y_i^2 - 2\tilde{\mu}_1 \, n\tilde{\mu}_1 + n\tilde{\mu}_1^2}{n - 1}$$
$$= \frac{\sum Y_i^2 - n\tilde{\mu}_1^2}{n - 1}$$

Bei sequentiell entstehender Stichprobe lassen sich $\sum y_i$ und $\sum y_i^2$ sequentiell aufsummieren; die Ergebnisse lassen sich anschließend unmittelbar angeben, während bei Verwendung von Form (3.2.15) erst nach Vorliegen der gesamten Stichprobe μ_1^* errechnet werden könnte und sich dann die Summierung von y_i-μ_1^* anschließen müßte.

(iii) Mit (i,ii):
$\sigma^{*2} = (750.46 - 30 \cdot 23.72) / 29$
$= 1.34$
$\sigma^* = 1.16$

(iv) Mit (3.2.21,3.2.22) unter Verwendung von σ^* für σ wird das Konfidenzintervall

$$(\mu_1^* - \sigma^*/\sqrt{3}\,,\ \mu_1^* + \sigma^*/\sqrt{3}) = (4.87 - 0.67\,,\ 4.87 + 0.67)$$
$$= (4.20\,,\ 5.54)$$

3.3.2: Sie könnten die jobs mit gewissen zeitlichen Initiierungsabständen starten, u.U. wiederholt, wenn der benchmark einen nur kleinen Umfang hat. Sie sollten darüber hinaus die Reihenfolge des Initiierens von jobs variieren, womöglich "zufällig" gestalten ("auswürfeln"). *3.3.2*

Leerseite