MAPLIS - Matrixorientierte Simulation, wieder aufgenommen

Wilfried Tettweiler, Krailling

wtettweiler@compuserve.com

ASIM 2014 - 22. Symposium Simulationstechnik, Berlin, 3.-5. September 2014

1 Vorwort

AZ münchen ★ Samstag/Sonntag, 14./15. November 1992 Seite 26

Keine Mark in der Kasse – aber sieben Millionen für die Wahrsager

Guter Rat ist teuer. Das wissen auch die Spitzenmanager im Münchner Rathaus und legen deshalb viel Geld an, wenn sie bei der Wahrheitsfindung selbst nicht weiter kommen. Die stolze Summe von sieben Millionen Mark ist im Haushaltsplan 1993 angesetzt, um die Rechnungen für Gutachter bezahlen zu können. Im Jahr 1990 kam die Stadt

Radlerkonzepts nichts wurde. Sie sollte 20 Millionen kosten.

Das renommierte „Prognos"-Institut aus Zürich hat sich Münchens Wirtschaftsstruktur vorgenommen. Das Gutachten ist gespickt mit einer Unmenge von Zahlen und Einzelanalysen. Am Schluß kommt raus: Eigentlich geht es München gar

Mit 200 000 Mark ist die Firma Schuh & Co aus Germering als Berater dabei. Sie führt im Auftrag des Planungsreferats „Verkehrszählungen im Bereich der Haupt-und Erschließungsstraßen" durch. Für die „vorbereitenden Untersuchungen für eine „Parkbevorrechtigung der Anwohner im Gebiet der südlichen Martiusstraße" gibt es

MAtrixorientierte Sprache für PLanspiel und Interaktive Simulation

1 Vorwort

mesoskopische Alternative zwischen eher mathematisch, technisch-physikalisch oder auch makroskopisch orientierten Mehrzweck-Simulationswerkzeugen

für in gängigen Simulationssprachen nicht Kundigen bleiben Modellzusammenhänge meist unverständlich, nicht nachvollziehbar und uninterpretierbar

2 Historischer Überblick

- FORTRAN-Bibliothek
- Vektorisierung
- Compiler

2.1 FORTRAN-Bibliothek (IBM 1975)

händische Umsetzung des Blockdiagramms in ein Modell durch Folgen von Aufrufen von Funktionen aus einer Laufzeitbibliothek

Variableninhalte in einem Virtuellen Speicher auf Festplatte gespeichert

dialogfähige Benutzerschnittstelle für Ablaufsteuerung, Datenein- und -ausgabe sowie das Ändern einzelner Werte

2.2 Vektorisierung (Cray 1980)

Laufzeiten nur noch Stunden statt Tage

CRAY-Übersetzer vektorisiert FORTRAN-Code der Laufzeitbibliothek automatisch, damit Parallelisierung bei der Ausführung

nur Batch-Ausführung, daher keine Dialogfähigkeit

2.3 Compiler (kommandoorientierte Programmiersprache 1990)

- Migration der Laufzeitbibliothek von FORTRAN nach PASCAL (Delphi)
- Modelldefinitionssprache und Benutzer-schnittstelle für Interaktion kommandoorientiert
- Baustelle!

3 Designidee

- Periodenbasierte Simulation
- Verringern des Statistikaufwandes
- Multidimensionale Rechenmethoden, insbesondere Aggregationsniveauwechsel unterstützen
- Modellsegmentierung

3.1 Periodenbasierte Simulation

Periodenwechsel markieren - nicht unbedingt äquidistante - Punkte auf der Zeitachse

bei Periodenwechsel werden alle Variableninhalte gespeichert und danach weiter verwendet

3.2 Vorauslaufenden Statistikaufwand reduzieren

Höhere statistische Verfahren (Regressions-analyse, Faktorenanalyse usw.) wegen der Informationsreduzierung vermeiden

dennoch Geschehensvielfalt in mehrdimen-sionalen Kontingenztafeln abbilden

daraus erhalten Bestandsgrößen absolute und Flussgrößen relative Häufigkeiten

3.3 Multidimensionale Daten verschiedener Aggregationsniveaus

Dimensionen (Indexe) beinhalten endliche Wertevorräte an Merkmalen

```
N of values    WE(2), KL(25), UE(45)
```

Variablendefinitionen basieren auf Dimensionen

```
Variable list SY, SZA (KL),
              S (KL) , E, SWE (KL by WE) ,
              SW, SE (KL),
              Q, DPXZA, SXZA, APXZA (UE)
```

Aggregationsniveaus werden durch die verwendeten Dimensionen festgelegt

3.3 Multidimensionale Daten verschiedener Aggregationsniveaus

Beispiel Bevölkerungsprognose:

Bestandsgröße Einwohnerbestand hat einen sehr hohen Detaillierungsgrad bzw. ein sehr niedriges Aggregationsniveau

Dimensionen sind Alter, Geschlecht, Nationalität, Wohntyp und Stadtviertel

109*2*2*3*429=561.132 Zellen

3.4 Aggregationsniveauwechsel

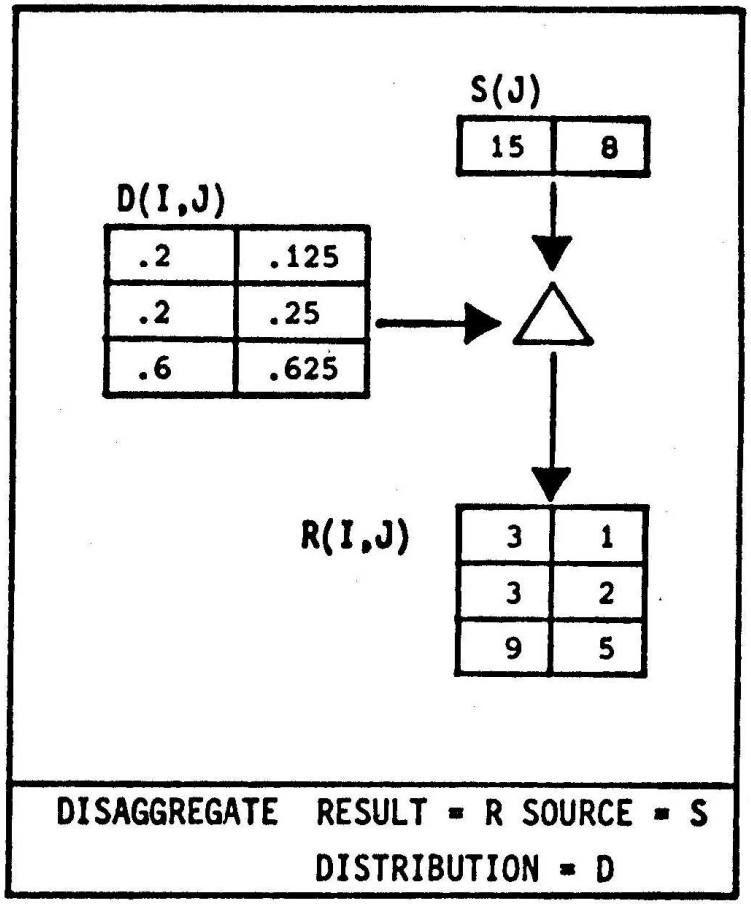
- Variablen mit gleichem Aggregationsniveau werden zellenweise verrechnet

Andernfalls:

- Disaggregieren mit Ganzzahloption
- Aggregieren
- Expandieren
- Ersetzen einer Dimension durch eine andere

3.4 Aggregationsniveauwechsel

Disaggregieren
bedeutet Hinzunehmen einer Dimension, deren Bestandsgrößen entsprechend den Anteilswerten der hinzugekommenen Dimension verteilt werden



3.4 Aggregationsniveauwechsel

Ganzzahloption (es gibt keine halben Kinder oder Zinsfüße mit vielen Nachkommastellen)

Modifizierte Monte-Carlo-Methode: Disaggregieren mit dem Ziel, einer vorgegebenen Verteilung möglichst nahe zu kommen.

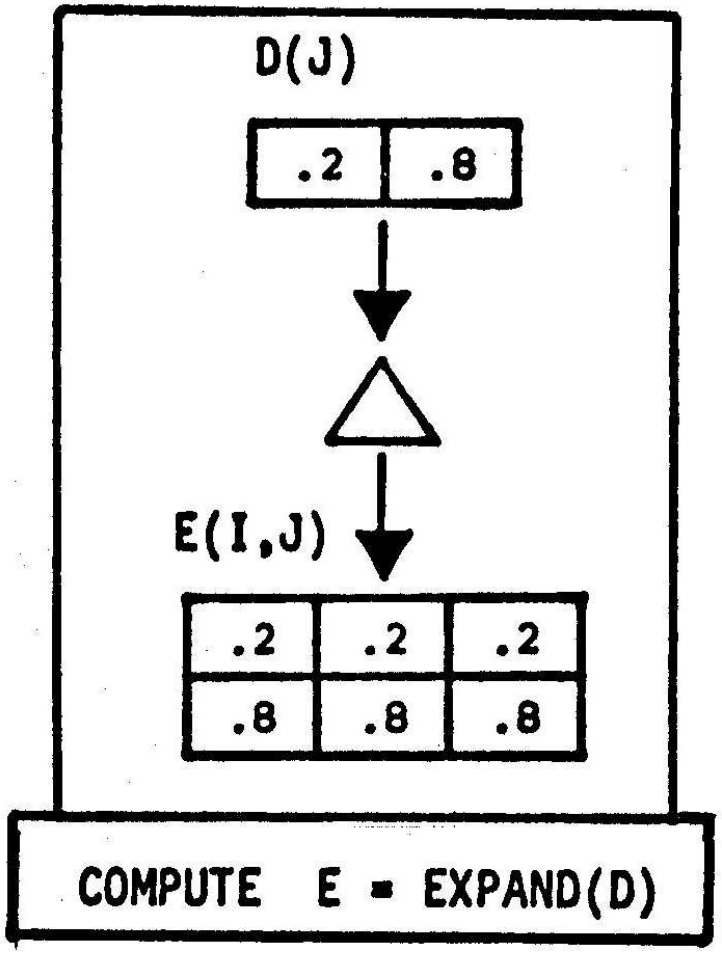
3.4 Aggregationsniveauwechsel

Aggregieren ist das Inverse zum Disaggregieren und bedeutet Wegfall mindestens einer Dimension durch Zeilen- oder Spalten-Summenbildung

eine Bestandsgröße erreicht ihr höchstmögliches Aggregationsniveau im (nulldimensionalen) Skalar mit der Totalsumme

3.4 Aggregationsniveauwechsel

Expandieren bedeutet Hinzunehmen einer Dimension, falls für unterschiedliche Aggregationsniveaus identische Häufigkeiten oder Wahrscheinlichkeiten gelten sollen



3.4 Aggregationsniveauwechsel

Ersetzen einer Dimension durch eine andere: die Option *PERMUTATION* berechnet auf der Basis entsprechender Zuordnungstabellen Zusammenfassungen von Merkmalsgruppen

Aggregieren von Alter zu Altersgruppen: ca. hundert Merkmale werden zu einigen wenigen zusammen gefasst und entsprechende Teilsummen gebildet

dynamische Wegenetze mit Übergangsmöglichkeiten: Schulkarrieremöglichkeiten zur Laufzeit durch Besetzen von Permutationsvektoren festlegen

3.5 Modellsegmentierung

Zerlegung in hierarchisch strukturierte Untermodelle

Abhängigkeiten zwischen Variablen werden jedoch nicht ausgewiesen

4 Die formale Sprache MAPLIS

- seit 1982 spezifiziert und mehrfach veröffentlicht
- um 1990 begonnen, auf PC zu implementieren
- in 1994 Tests mit ZORAN Vektorprozessor
- Tests bis heute ausgesetzt

4.1 Spracheigenschaften

prozedurale Sprache ohne echtes Timing

kennt ausschließlich den Datentyp Float, daher keine Datentypprüfung erforderlich

Verträglichkeit der Aggregationsniveaus wird bereits zur Übersetzungszeit geprüft

Laufzeitbibliothek enthält Standardmethoden aus Mathematik und Statistik für Zuweisungen und Berechnungen im Modell, ist erweiterbar

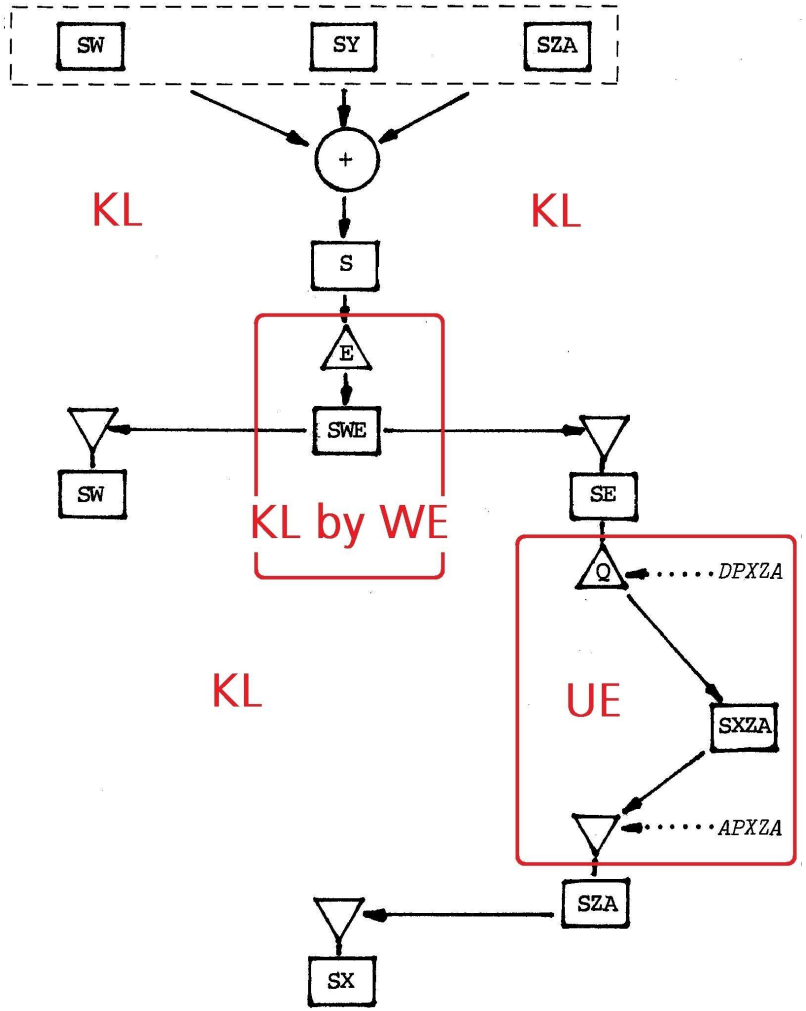
4.1 Spracheigenschaften

Syntax hat die Backus-Naur-Form für den Einsatz eines Compiler-Compilers:

```
maplis ::= { ( ( declare_section | run_section | info ) [ ! ] ) | include } finish [ ! ]
declare_section ::= declare
declare ::= declare_values | declare_vars | declare_models
declare_values ::= n_of_values | value_labels
n_of_values ::= N_OF_VALUES i_def { i_def }
i_def ::= indexname ( ( [ valuelist ] ) | ( ( number ) ) )
value_labels ::= VALUE_LABELS one_index_value_l { / one_index_value_l }
one_index_value_l ::= indexname ( value ) label { ( value ) label }
declare_vars ::= add_variables | print_formats | var_labels
add_variables ::= ( ADD_VARIABLES | VARIABLE_LIST ) varlist ( [ indexlist ] ) [ = constant ]
print_formats ::= PRINT_FORMATS varlist ( width [ , decimals ] )
var_labels ::= VAR_LABELS var_label { / var_label }
var_label ::= varname label
declare_models ::= add_model body end_model
add_model ::= ADD_MODEL modname [ label ] [ ! ]
end_model ::= END_MODEL [ label ]
body ::= statement { statement }
statement ::= b_statement | d_statement
b_statement ::= aggregate | compute | disaggregate | let
aggregate ::= AGGREGATE RESULT = varname SOURCE = varname [ PERMUTATION = varname ] [ DISTRIBUTION = varname ]
compute ::= COMPUTE varname = expression
expression ::= term { ( + term ) | ( - term ) }
term ::= factor { ( * factor ) | ( / factor ) }
factor ::= varname | functioncall | ( ( expression ) )
functioncall ::= call_s | call_s_s | call_s_n | call_s_i_n | call_s_i | call_s_v
call_s ::= fctname_s ( varname )
fctname_s ::= ( NOT | SQRT | LN | EXP | SIN | COS | TAN | ATAN | SUM | EXPAND | INV | TRANS )
call_s_s ::= fctname_s_s ( varname , varname )
fctname_s_s ::= ( AND | OR | XOR | EQ | NE | GT | LT | LE | GE | MAX | MIN )
call_s_i_n ::= fctname_s_i_n ( varname , indexname , number )
fctname_s_i_n ::= SHIFT
call_s_n ::= fctname_s_n ( varname [ , number ] )
fctname_s_n ::= ( LAG | MOD )
call_s_v ::= fctname_s_v ( varname , value )
fctname_s_v ::= SUB
call_s_i ::= fctname_s_i ( varname , indexname )
fctname_s_i ::= ( SORTA | SORTD )
disaggregate ::= DISAGGREGATE RESULT = varname SOURCE = varname DISTRIBUTION = varname [ PERMUTATION = varname ]
d_statement ::= call_model | exit | if | interaction | return
call_model ::= CALL_MODEL modname
exit ::= EXIT [ label ]
if ::= IF ( varname ) ( call_model | interaction | exit | return | compute | let ) FI
interaction ::= INTERACTION [ label ]
return ::= RETURN [ label ]
run_section ::= data_definition | let | run_model | write_var
data_definition ::= file_interface | read_var
file_interface ::= inout_format | inout_medium
inout_format ::= INOUT_FORMAT one_inout_format { , one_inout_format }
one_inout_format ::= varlist fmtdef
inout_medium ::= INOUT_MEDIUM one_inout_medium { , one_inout_medium }
one_inout_medium ::= varname ( streaminout | fileinout )
streaminout ::= { , varname } CARD
fileinout ::= dsname
read_var ::= read_data
read_data ::= READ_DATA varlist
fmtdef ::= ( ( fixfmtdef | varyingfmtdef | internfmtdef | valuesfmtdef ) )
fixfmtdef ::= F width [ , decimals [ , repetitions [ , skipcols ] ] ]
varyingfmtdef ::= V [ skipcols ]
internfmtdef ::= I
valuesfmtdef ::= indexname
width ::= number
decimals ::= number
repetitions ::= number
skipcols ::= number
let ::= LET varname ( { value } ) = ( constant | varname )
run_model ::= RUN_MODEL modname ( ( until_cond ! { ( dialogue | continue ) } ) | ! ) stop
until_cond ::= UNTIL [ SEQNUM = number ] [ SUBMODEL = modname ( ENTRY | INTERACTION | RETURN ) ] [ INTERACTION ] [ RETURN ]
dialogue ::= ( let | write_var ) [ ! ]
continue ::= CONTINUE [ until_cond ] !
stop ::= STOP !
write_var ::= print | write_data
print ::= PRINT varlist [ NOVARLABEL ] [ NOVALLABEL ]
write_data ::= WRITE_DATA varlist
include ::= INCLUDE dsname !
info ::= INFO [ NOVARLABEL ] [ NOVALLABEL ] ]
finish ::= FINISH
number ::= int
constant ::= real
dsname ::= string
value ::= valueid | number
valueid ::= ident
valuelist ::= value_range { , value_range }
value_range ::= value [ . . value ]
indexlist ::= indexname { ( , | BY ) indexname }
indexname ::= ident
label ::= string
modname ::= ident
varlist ::= varname { , varname }
varname ::= ident
```

4.2 Syntaxbeispiel

Blockdiagramm eines einfachen Schülerdurch-flussmodells für einen einzelnen Schulstandort



4.2 Syntaxbeispiel

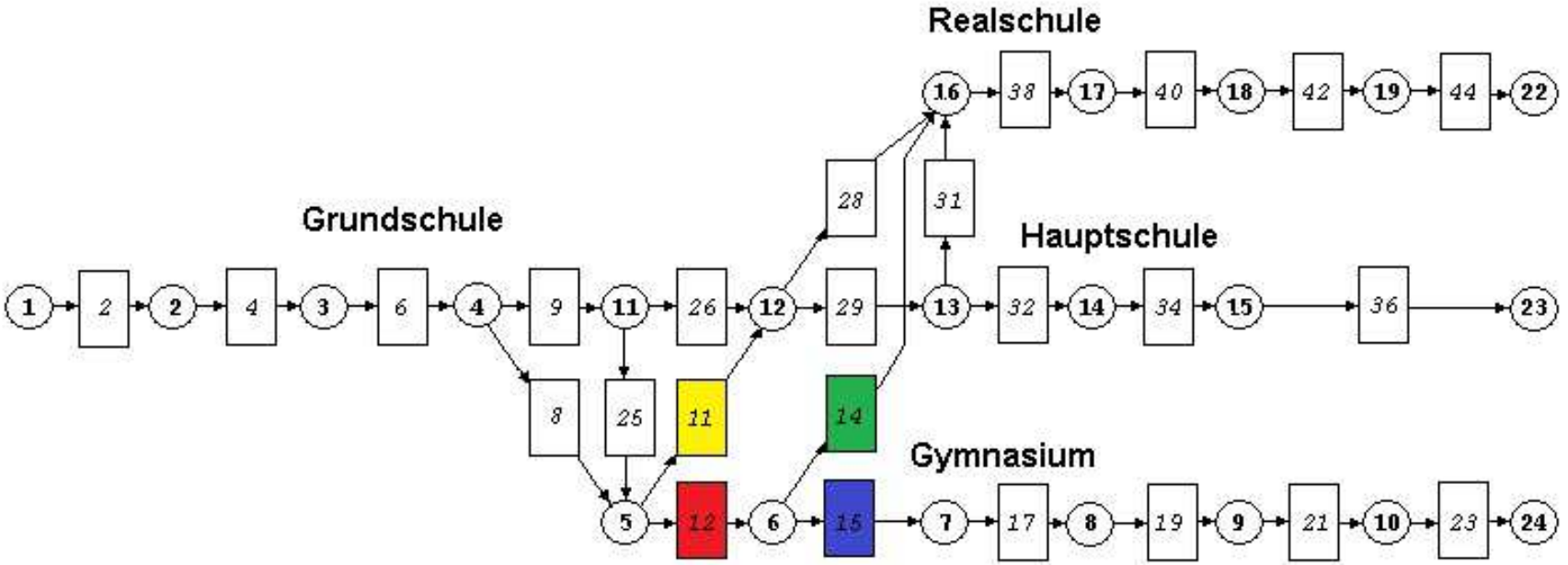
Deklarationsteil

(MAPLIS ähnelt in seiner Nomenklatur der Kommandosprache von SPSS)

```
Run name        Schülerdurchfluß simulieren
Comment         nachfolgend Deklarationen der Dimensionen (Indizes)
                und darauf basierend der Variablen
N of values     WE(2), KL(25), UE(45)
Value labels    WE (1) Klassenziel nicht erreicht (2) erfolgreich /
                KL (4) 4. Jgst. Grundschule
                   (5) 5. Jgst. Gymnasium
                   (10) 10. Jgst. Gymnasium
                   (11) 5. Jgst. Hauptschule
                   (22) Abschlüsse Realschule
                   (23) Abschlüsse Hauptschule
                   (24) Mittl.Reife Gymnasium
                   (25) Abgänge-Wegzüge
Variable list   SY, SZA (KL),
                S (KL) , E, SWE (KL by WE) ,
                SW, SE (KL),
                Q, DPXZA, SXZA, APXZA (UE)
Var labels      SY Zuziehénde Schüler /
                SZA Vorrückende bzw. übertretende Schüler /
                S Schüler insgesamt /
                E Erfolgswahrscheinlichkeit /
                SWE Nach Erfolg getrennte Schüler /
                SW Wiederholende Schüler /
                SE Erfolgreiche Schüler /
                Q Übertrittswahrscheinlichkeit /
                DPXZA Herkunftsmerkmal nach Übertrittsart /
                SXZA Schüler nach Übertrittsart getrennt /
                APXZA Zielmerkmal nach Übertrittsart /
```

4.2 Syntaxbeispiel

Wegenetz der Übergangsmöglichkeiten



4.2 Syntaxbeispiel

Wegenetz kann dynamisch geändert werden

bis zu 45 Klassenübergänge sind durch die mit UE dimensionierte Verteilung Q und das Permutationsvektorpaar DPXZA (Quellen) und APXZA (Ziele) festgelegt

	1	2	3	4	5	6	7	8	9	
Q	3,00%	97,00%	0,50%	99,50%	0,50%	99,50%	0,50%	23,50%	76,00%	. . .
DPXZA	1	1	2	2	3	3	4	4	4	
APXZA	25	2	25	3	25	4	25	5	11	

10	11	12	13	14	15	16	17	18	
2,80%	6,90%	90,30%	2,00%	5,00%	93,00%	3,00%	97,00%	3,00%	. . .
5	5	5	6	6	6	7	7	8	
25	12	6	25	16	7	25	8	25	

4.2 Syntaxbeispiel

Modelldefinitionsteil

```
Add model       SDFSIM Schülerdurchfluß-Simulationsmodell
Document        Vorgehensweise: Schüler verteilen auf Vorrückende und
                Wiederholende, Vorrückende verteilen entsprechend der
                Übertrittsart, Zusammenführen zu neuen Klassenverbänden
                und Hinzufügen der Wiederholer.
Disaggregate    Result = SWE Source = S Distribution = E
Compute         SE = sub (SWE,2)
Compute         SW = sub (SWE,1)
Comment         Wiederholer werden erst später wieder gebraucht, nun
                entsprechend Übertrittswahrscheinlichkeiten auf die
                verschiedenen Alternativen verteilen
Disaggregate    Result = SXZA Source = SE Distribution = Q Option = INT
                Permutation = DPXZA
Aggregate       Result = SZA Source = SXZA Permutation = APXZA
Compute         S = SW + SZA
End model
```

4.2 Syntaxbeispiel

Testlauf des Modells

Die Schüler sind um eine Klasse vorgerückt und es wurden Anzahlen für Abschlüsse berechnet.

```
Read data          S, E, Q, DPXZA, APXZA
1082,1095,1109,1154,353,368,371,366,369,365,971,997,771,766,737,
351,410,422,393/
*** WARNING: VARIABLE S          INCOMPLETE DATA ***
.04,.03,2*.02,.04,.07,.14,.14,.15,.12,5*.01,.04,2*.16,.08,6*0,
.96,.97,2*.98,.96,.93,.86,.86,.85,.88,5*.99,.96,2*.84,.92,6*0/
*** MESSAGE: VARIABLE E      DATA ACCEPTED ***
.030,.970, .005,.995, .005,.995, .005,.235,.760,
.028,.069,.903, .02,.05,.93, .03,.97, .03,.97, .01,.99, .01,.99,
.003,.051,.946, .002,.237,.761, .017,.072,.911, .093,.907, .01,.99,
.001,.999, .025,.975, .007,.993, .055,.945/
*** MESSAGE: VARIABLE Q      DATA ACCEPTED ***
2*1, 2*2, 2*3, 3*4, 3*5, 3*6, 2*7, 2*8, 2*9, 2*10, 3*11, 3*12, 3*13,
2*14, 2*15, 2*16, 2*17, 2*18, 2*19/
*** MESSAGE: VARIABLE DPXZA  DATA ACCEPTED ***
25,2, 25,3, 25,4, 25,5,11, 25,12,6, 25,16,7, 25,8, 25,9, 25,10, 25,
24, 25,5,12, 25,16,13, 25,16,14, 25,15, 25,23, 25,17, 25,18, 25,19,
25,22/
*** MESSAGE: VARIABLE APXZA  DATA ACCEPTED ***
Run model      SDFSIM Seqnum = 1982 Entry
*** MESSAGE: MODEL SDFSIM, PERIOD 1982, INTERRUPT AT ENTRY ***
*** MESSAGE: MODEL SDFSIM, PERIOD 1982, INTERRUPT AT RETURN ***
Print            SZA
--> KL
+-----+
I SZA I  Vorrückende bzw. übertretende Schüler
+-----+
```

1	2	3	4	5	6	7	8
			4. Jgst. Grundschu	5. Jgst. Gymnasium			
0.	1008.	1057.	1081.	315.	306.	318.	309.

9	10	11	12	13	14	15	16
	10. Jgst. Gymnasium	5. Jgst. Hauptschu					
305.	313.	859.	933.	751.	695.	688.	306.

17	18	19	20	21	22	23	24
					Abschluss e Realsch	Abschluss e Hauptsc	Mittl.Rei fe Gymnas
337.	336.	352.	0.	0.	342.	727.	321.

25
Abgänge-W egzüge
206.

```
Stop
*** MESSAGE: MODEL SDFSIM, PERIOD 1982, STOP AT RETURN ***
Finish
```

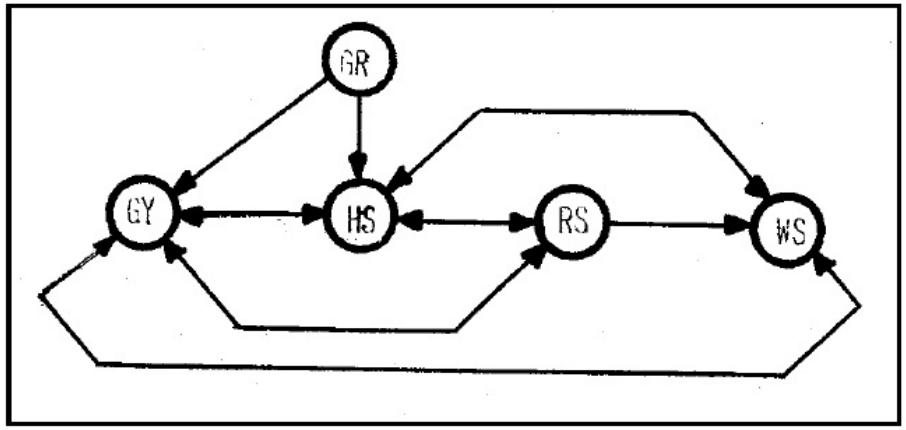
5 Modellentwürfe und -anwendungen

- Schülerprognose: Staatsinstitut für Bildungsforschung und Bildungsplanung, München (1974-1978)
- Bevölkerungsprognose: Planungsreferat der Landeshauptstadt München (1978-1993)

Auskünfte über Modelldetails kann man bei den jeweiligen Verfassern erfragen!

5.1 Schülerprognose

Schularten Grund-, Haupt, Real- und Wirtschaftsschule sowie Gymnasium mit den Dimensionen Schularten, Übertrittsquoten, Schulstandorte und innerbayerische Migration



5.1 Schülerprognose

weitere Dimension Schulstandort: Permutationstabellen entsprechend jeweils vertretenen Schularten angepasst

Migration zwischen Schulstandorten: relativ schwach besetzte Übergangsmatrix, meist nur Zellen besetzt für Flüsse der Art "verbleibt am Schulstandort" oder "wechselt an einen anderen Schulstandort, weil nur dort die entsprechende weiterführende Schulart existiert"

abgespeckter Modelllauf: statt 3000 Schulstandorten lediglich ca. 70 Landkreise und 25 kreisfreie Städte

5.2 Bevölkerungsprognose

"Münchner Simulationsmodell"

mehr als 30 Untermodelle für Bevölkerungsentwicklung, Bautätigkeit, Erwerbspersonen, Arbeitstättenplanung und Schulraumplanung

Dialogfähigkeit: Möglichkeit, im laufenden Modell Eingriffe in die Datenstruktur entsprechend planerischer Zielsetzungen vorzunehmen

5.2 Bevölkerungsprognose

kartografische Ergebnisdarstellung: Allokationsmodell zur Ermittlung des Versorgungsgrades der Stadtviertel

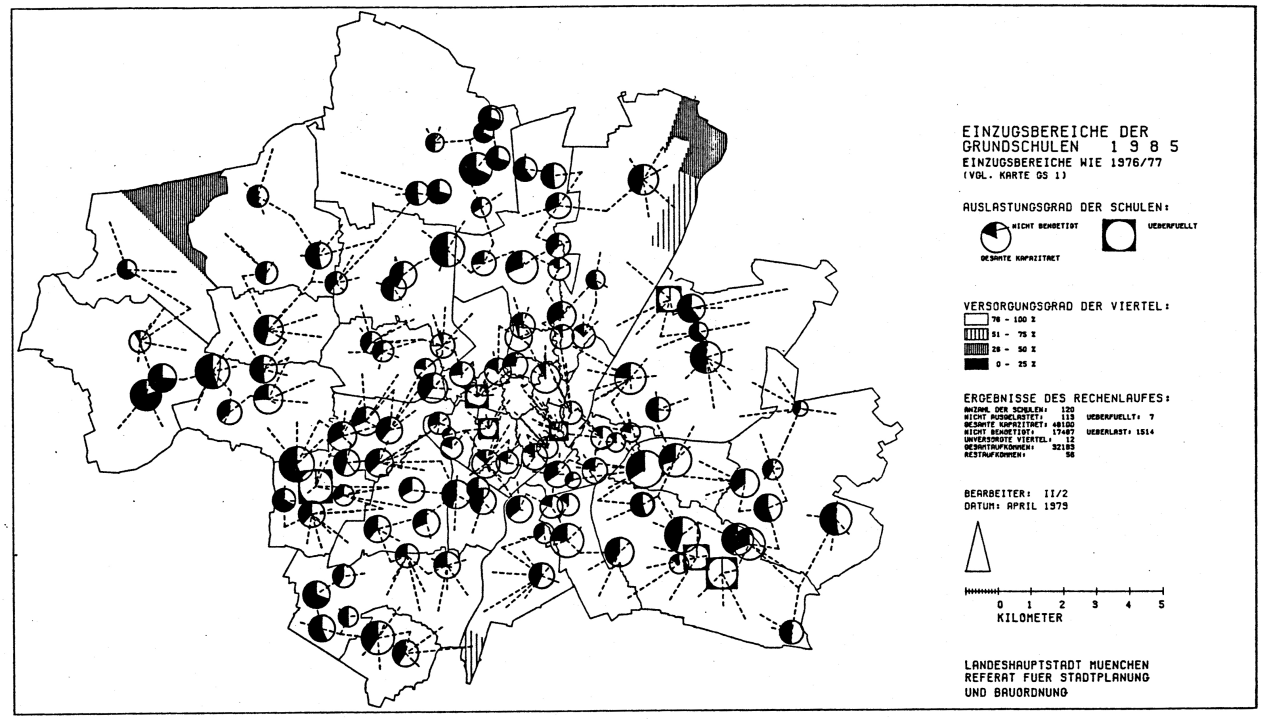


Bild 2: Auslastung der sozialen Infrastruktur (GS 2)

6 Ausblick

- durch PCs weniger technische Begrenzt-heiten als vor zwanzig Jahren beim Main-frame-basierten Rechenbetrieb mit Zwang zur sequentiellen Verarbeitung, Mangel an Speicherplatz und Rechenzeit
- fortlaufend verbesserte Realisierungschancen auf PCs

6.1 Moderne Hard- und systemnahe Software

Prozessorkernanzahl und -geschwindigkeit

Speicher größer, schneller und erschwinglicher, intern (RAM) und extern (Solid State Disks).

Datenaustausch auf platzsparenden und preiswerten Transportmedien (USB-Platten und -Sticks), E-Mails, Zugriff auf die Cloud

Parallelisierung durch Nutzung von Grafikprozessoren (GPUs, siehe Parallel Computing Toolbox von MathWorks, Intel Parallel Studio XE oder bei den Nvidia CUDA-Fortran-Werkzeugen)

6.1 Moderne Hard- und systemnahe Software

PC-Betriebssysteme (64bit) mit größerem Adressraum für große Matrizen

leistungsfähige Datenbanksysteme

erleichtertes Synchronisieren von simulierten mit realen Daten dank standardisierter Schnittstellen (Quasistandards MS Excel, xBASE)

Und die Preisentwicklung bleibt weiter erfreulich!

6.2 Visuelle Programmierumgebung

Menü- und Icon-basiertes grafisches Interface zur Modelldefinition wünschenswert

darin farbliche Gestaltung von Flächen, die Variablen gleichen Aggregationsniveaus enthalten

automatische Umwandlung der grafischen in eine skriptbasierte Modelldarstellung und umgekehrt

6.3 Verbesserungen für Eingabe und Ausgabe

Anzahl vorgehaltener Dateischnittstellen vermehren

integrierte Schnittstelle zu einem GIS-System zur Darstellung von Simulationsergebnissen mit geografischem Bezug

Business Intelligence (BI) zur Auswertung und grafischen Darstellung multidimensionaler Simulationsergebnisse

6.4 Datenbankschnittstelle

auch für das Speichern von Variableninhalten beim Periodenwechsel, von schwach besetzten Matrizen die Konzepte neuer Datenbanken nutzen

strukturierte Datenbanken (noSQL), Multidimensional Clustering und Multidimensional Expressions (MDX)

Ich danke für Ihre Aufmerksamkeit!