

# USING COURSE CONSTRAINTS IN SIMULATING STUDENT POPULATIONS

January 25, 2007

**THANASSIS HADZILACOS<sup>1,2</sup>, DIMITRIS KALLES<sup>1</sup>, DIMITRIS KOUMANAKOS<sup>3</sup>**

<sup>1</sup>Hellenic Open University, Sachtouri 23, 26222, Patras, Greece

<sup>2</sup>Research Academic Computer Technology Institute, University of Patras Campus, Rio, Greece

<sup>3</sup>Department of Mechanical and Aeronautical Engineering, University of Patras, Rio, Greece

Emails: {thh, kalles} @ eap.gr, kouman @ upatras.gr

Contact: [kalles@eap.gr](mailto:kalles@eap.gr)

## ABSTRACT

Distance learning universities usually afford their students the flexibility to advance their studies at their own pace. This can lead to a considerable fluctuation of student populations within a programme's courses. The evolution of the student population may be an important factor in determining the academic viability of a programme as well as the resources that have to be budgeted and administered. Providing a method that estimates this population could be of substantial help to university management and academic personnel. We describe how course precedence constraints expressed in the Prolog language are used to calculate alternative tuition paths and we then use discrete Markov simulation with the Extend system to estimate future populations. In doing so, we identify key issues of the potential deployment of such a system at a large scale.

## KEYWORDS

Simulation, population estimation, precedence constraints, educational planning.

## 1. Introduction

Distance learning universities usually afford their students the flexibility to advance their studies at their own pace. This can lead to considerable fluctuation of student populations within a programme's courses, possibly affecting the academic viability of a programme as well as the resources that have to be budgeted and administered. Providing a method that could guide management and academic personnel towards estimating this population could be of substantial administrative value (Chang and Radi, 2001; Webster, 1997).

Such fluctuations also occur in the Hellenic Open University<sup>1</sup> where students' personal circumstances may easily change within short periods, mostly due to family and employment reasons. Moreover, as in most similar universities (Open Learning, 2004), significant drop-out rates are recorded in some programmes, usually as a result of failure in a junior year. While understanding and addressing the reasons of failure is an educational problem, drop-out also amplifies the administrative consequences of unexpected fluctuations in the student population.

---

<sup>1</sup> HOU was founded in 1992 and offered its first courses in 1998. Currently, over 25,000 students (mostly adult) are enrolled and over 1,500 tutors are active on a yearly basis.

In HOU, administrative aspects that are mostly affected by the student population include tutor contract renewal, tutoring venue rental and allocation, the procurement and distribution of educational material, and the development and operation of (mostly IT) infrastructure. As only about 15% of HOU running costs are provided by the government and the rest is borne by students, cost consciousness is essential during planning and before taking decisions, especially so if one might consider venturing into several-year contracts.

In this paper we present a method to estimate student populations based on course precedence constraints, as stipulated by programme regulations. These constraints are used to calculate alternative tuition paths for students using discrete simulation, based on data about past enrolments and exam successes (note that no individual records are examined).

The rest of the paper is structured in six sections. We next offer some more details on the educational setting at HOU, mainly to introduce the associated nomenclature. We then present the specification and the results of the simulation, in two separate sections. Following that we backtrack to the simulation specification and show how it can be computed from higher level data, to alleviate the error risks inherent in the manual development of large models. Then, we identify the issues that we need to resolve before we field our approach at a larger scale. Finally, while concluding, we also briefly reflect on the political aspect of using simulation for educational planning.

## **2. Background on the Educational Application Field**

In this work we have focused on a Master's conversion programme in Information Systems offered at HOU. Students have the opportunity to acquire specialized knowledge in Information and Communication Technologies, and to prepare for professional work in the design, development and management of integrated information systems. The programme is targeted at science and engineering graduates and covers the design and development of software and systems, the management and the quality of system development, and advanced issues in telecommunications and networking. It offers five taught modules, four of which must be completed to proceed to a thesis. Of those modules, one is a compulsory and demanding introduction to the programme with recorded success rates of about 50-70% and drop-out being the usual path after failure.

A module is the basic educational unit at HOU. It runs for about ten months and is the equivalent of about 3-4 conventional university semester courses. A student may register with up to three modules per year. For each module, a student is expected to attend five plenary class meetings throughout the academic year. A typical class contains about thirty students and is assigned to a tutor (tutors of classes of the same module collaborate on various course aspects). Class meetings are about four hours long and are structured along tutor presentations, group-work and review of homework. Furthermore, each student must turn in some written assignments (typically four or six), which contribute towards the final grade, before sitting a written exam.

Students fail a module and may not sit the written exam if they do not achieve a pass grade in the assignments they turn in; these students must repeat that module afresh. A student who only fails the written exam may sit it on the following academic year; such students are also assigned to student groups but the tutor is only responsible for marking their exam papers.

In earlier studies, Xenos *et al.* (2002) reported that undergraduate students who dropped out usually claimed that they were not able to correctly estimate the time that they would have to devote to their professional activity and, as a result, the time dedicated to their education decreased unexpectedly. Some also felt that their knowledge was not sufficient (other reasons, such as family or health issues were also quoted). Our experience with a demanding postgraduate programme suggests that these reasons apply as well.

We have to date invested on a separate research direction to analyse students' performance in various modules, based on assignment and exam data. In this paper, we describe a top-down approach that aims to estimate student populations based on historical enrolment data; eventually we will attempt to demonstrate that it is complementary to the other direction. To put it in a nutshell, we believe that educational intelligence must be built upon data drawn from various sources and at varying resolutions; it is up to academic personnel to integrate the views and exploit the findings with concrete decisions.

It is interesting to note that the methodology developed in this paper was first conceived to address the problem of estimating the number students in the programme, so as to better argue about increasing the number of students that can be admitted at registration. Such a direction would not be

easy to explore unless one has a solid appreciation of the demand for the programme as well as an appreciation of the potential availability of suitable tutoring personnel.

### 3. Discrete Simulation - Specification

Before embarking on a discrete simulation we must design the state space and the state transition probabilities. A suitable state for our experiment is the set of courses that a student has selected. Figure 1 shows a *very* small but typical extract of such a design where numbers indicate course codes (transition probabilities are not shown, to avoid cluttering). The interpretation is as follows: at start the student may either select one or two courses (a student cannot select course 51 on its own as a first course), upon successful completion of which he may then choose again one or two courses. Note the italicised number annotating a transition line: it conveys the information that the transition happens upon selection of the particular course.

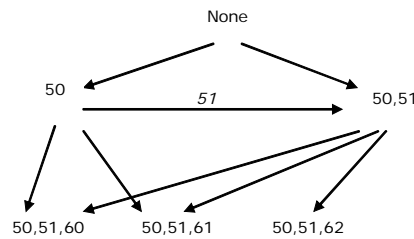


Figure 1: A visual model of precedence constraints.

The above model uses transition labels to convey information on which courses a student is currently attending. In our design we have used a more verbose notation, with equivalent representational power but easier to comprehend by humans as regards course selection. Figure 2 further elaborates part of Figure 1 as an example. Note that what used to be *state:50,51,61* in Figure 1 has now been broken in two states, correspondingly increasing the number of transitions fanning out from (what used to be) *state:50,51*. The new notation uses a *dash* (-) to separate completed courses from currently attended courses. So, with reference to Figure 2, we note that a student who has completed unit 50 and is currently attending 51 (see encircled state), may be successful and then register for unit 51 (left branch) or may fail and repeat unit 51 along with a new registration for unit 61.

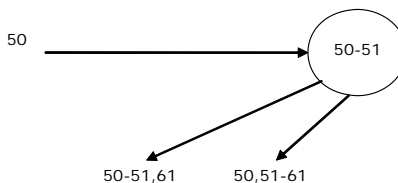


Figure 2: A verbose variant of the visual model of precedence constraints.

The full representation must account for a range of possibilities on selecting more than a course per year, on deciding the order of course selection (when that is an option), or on deciding which optional courses to select. For the postgraduate programme we are studying, where five taught courses are offered and four of them must be completed to proceed to a thesis, this amounts to a space of several dozens of states and transitions. Representation size may scale into hundreds of graph elements for a programme of several units. Moreover, that space must be enhanced with transitions from every state to itself (to account for failing a course or a combination of courses and having to repeat it), as well as with transitions from every state to a sink state (to account for those who drop out of their studies).

The specification of the transition probabilities for our postgraduate programme was based on the statistics of the first year of the programme's running. For the senior year modules where data on success and failure were not available we extrapolated a value based on our tutoring experience (based on the observation that senior year students are far less likely to fail).

We have used the *Extend*<sup>2</sup> simulation software to code the discrete simulation experiment. The Extend software belongs to the visual programming genre where the programmer selects from a library of modules (components, functions) and creates flows between them; these flows then serve as pathways for discrete simulation objects (simulated students in our case) to move around into the simulated system. Figure 3 shows a snapshot of the Extend code that implements our experiment; we note that the shown extract accounts for about one-fifth of the overall program.

To briefly guide the reader around the Extend concepts, we now coarsely describe some graphical programming elements (from left to right). We first note two identical lines running parallel to each other that mostly contain the simulation logic in terms of randomized experimentation, look-up tables for transition probabilities, etc. (that structure is replicated at other parts of the code). Immediately to the right of those visual structures one can see a vertical group of about a dozen rectangular blocks; these are object counters which tally the number of (simulated) students arriving at a particular state of the state space graph. Note that the output of such a block (see the arrowed lines for an example) is directed to a log file (rightmost column of three elements) and to a state to serve as input for the next time unit (year).

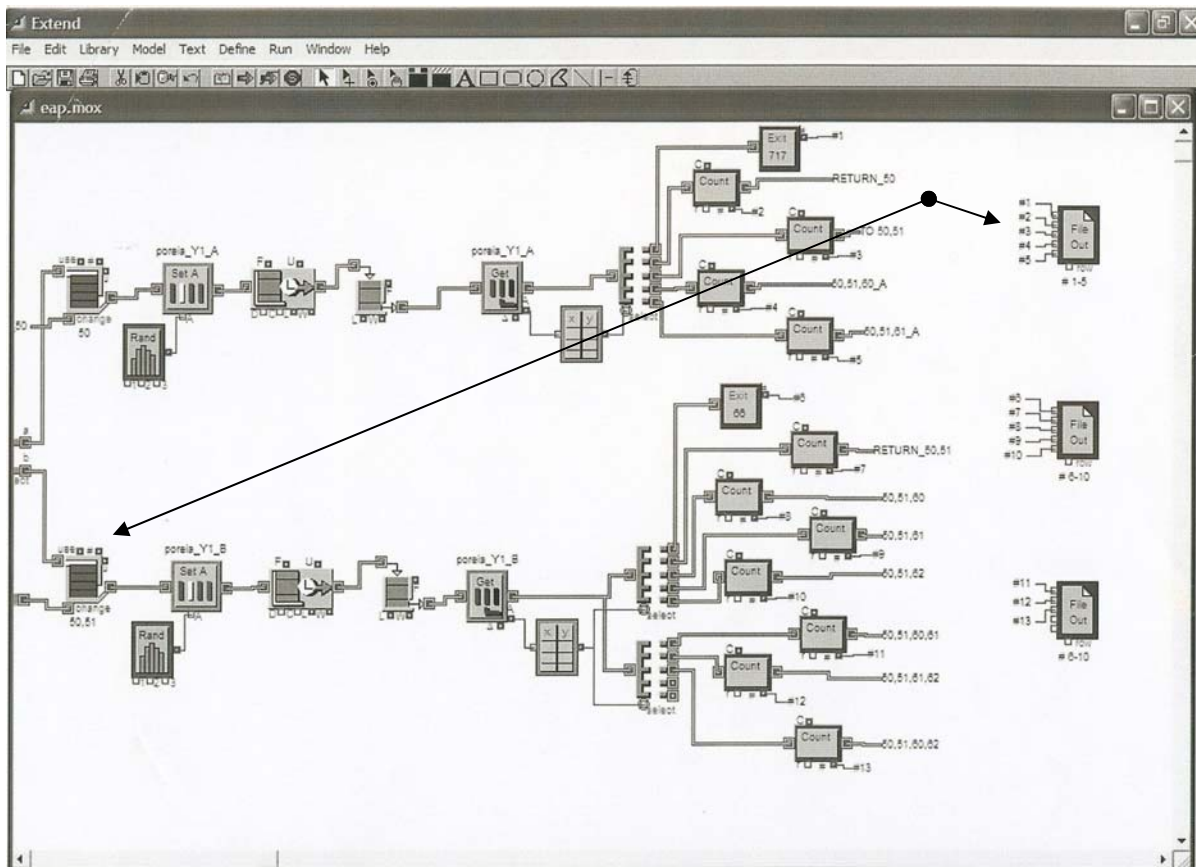


Figure 3: A snapshot of the Extend code that implements our experiment.

#### 4. Discrete Simulation - Results

We have conducted experiments with the above implementation to provide a proof of concept *and* to obtain estimates for the actual student population of the postgraduate programme that we are running. For the transition probabilities we have used the values that best reflected the actual statistics after the first year of the programme's running and we experimented with nine different probability configurations that we thought would provide us with an appreciation of the variability of the population estimates. We developed these configurations by deciding to only vary some

<sup>2</sup> *Extend* is available from <http://www.imaginethatinc.com/>.

probabilities that were deemed to be important all-around the simulation. It goes without saying that in an operational context, this is a task that has to be well designed and carried out.

In those configurations the following transition probabilities were focused on, which corresponded to the most easily observable student paths in the programme:

- At the outset of the programme, a student may start with enrolling either in unit 50 or in units 50 and 51. The probability to only select unit 50 was set at ( $P_A$ ) 0.6, 0.65 or 0.7.
- There is a relatively high probability to fail unit 50 and to drop out altogether subsequently, which was set at ( $P_B$ ) 0.5, 0.4 or 0.3.
- A successful completion of unit 50 is followed by an obligatory enrolment in unit 51 and an optional enrolment in a second year unit, usually unit 60.
  - o The probability to select those two units was set at ( $P_C$ ) 0.15, 0.2 or 0.25.
  - o The probability to select just one unit was set at ( $P_D$ ) 0.25, 0.3 or 0.35.

All other transitions from any state were assigned default probabilities, by subtracting all probabilities that reflected the special cases, by allowing for a small drop out probability at senior units, and by assigning values based on the actual statistics of the previous real population.

For the above mentioned probabilities we used the combinations shown in Table 1:

Table 1. Probability combinations for experimental batches

# Exp	$P_A$	$P_B$	$P_C$	$P_D$
1	0.60	0.50	0.25	0.15
2	0.60	0.40	0.30	0.20
3	0.60	0.30	0.35	0.25
4	0.65	0.50	0.25	0.15
5	0.65	0.40	0.30	0.20
6	0.65	0.30	0.35	0.25
7	0.70	0.50	0.25	0.15
8	0.70	0.40	0.30	0.20
9	0.70	0.30	0.35	0.25

While an exhaustive combination of the above mentioned probabilities would probably lend more credibility to the overall experiment, our decision to carry out a smaller scale of experiments was coupled with combining those probabilities in a way that would allow the simulation to produce larger differences in the estimated populations, thus also allowing us to obtain some insight into the robustness of the estimates.

Each configuration was used as an input to an experimental batch that simulated 20 academic years with 120 new students enrolling each academic year (this number has now increased to 150, but we defer to Section 6 the discussions about the potential for a real deployment). We then averaged the populations enrolled in each unit for the years and report the averages. Since we report all experiments together, we have opted to not include deviations in our results to avoid cluttering; still, one should be able to derive from the presentation of quite different experiments that deviations within a single experiment would have to be smaller than differences across experiments.

The estimated populations are shown in Table 2:

Table 2. Estimated student populations per unit

# Exp	50	51	60	61	62
1	156	68	61	28	51
2	151	62	55	26	46
3	148	60	52	23	45

<b>4</b>	153	72	63	28	54
<b>5</b>	151	67	55	28	50
<b>6</b>	144	62	51	25	46
<b>7</b>	149	74	65	28	55
<b>8</b>	148	72	57	30	55
<b>9</b>	144	67	51	27	50
<b>Min</b>	144	60	51	23	45
<b>Max</b>	156	74	65	30	55

Briefly commenting on these results we note that there is relative stability in the measured results if probabilities are allowed to fluctuate within a reasonable range. A strictly operational interpretation would suggest that those numbers might make a difference of at most one student group per unit, thus requiring the recruitment of more tutors (note that since students are dispersed in major cities, the result might well be just larger groups per city).

## 5. A (Visual) Grammar for Generating the Discrete Simulation Model

The space of student states, along with numbers on enrolment and transition probabilities, is a complete specification for the simulation experiment. However, the transitions simply reflect the rules of an academic program. Undergraduate programs can accommodate more than a dozen courses; for those, the complete specification may be simply unmanageable to draw and the process is prone to errors. A reasonable extension is a modelling notation that precisely captures the precedence constraints between courses and can automatically generate transitions. Such or visual model is shown as an example in Figure 4.

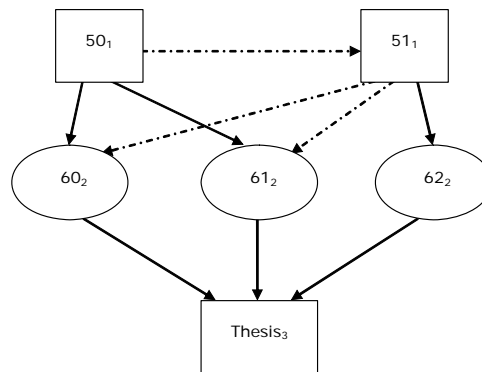


Figure 4: A visual model of precedence constraints.

In the figure above, numbers indicate course codes and subscripts indicate the nominal year of the programme where a course unit has been allocated. A solid line indicates a hard precedence, in the sense that a unit cannot be selected (for enrolment) unless its precedent has been completed. A dashed line indicates a soft precedence, in the sense that a unit cannot be selected unless its precedent has at least started. A rectangle indicates a compulsory unit; an oval indicates an optional one.

A key advantage of the above notation is that it is easy to communicate to users who may be non-versed in such formalisms. While it may not be straightforward to request that they produce the model, it should nevertheless be at least easy to convince about its correctness. Note that the precedence constraints can be derived from regulations and are thus not subject to subjective interpretations.

The above visual tool can be thought of as a prototyping tool. To develop the full space of available transitions, some other features may be necessary. Drawing on our experience we have noted that

optional characterizations of a course unit may be whether it must be among the first where the students enrol (as is the case with foundation courses) or, whether it may be the last (as is the case with theses), or whether a latter year unit may or may not be selected unless all previous year units have been selected.

We have elected the Prolog programming language to capture these specifications. Table 3 shows a code extract describing some constraints of our IS program.

Table 3. Fundamental IS program constraints

---

```
units([pls50,pls51,pls60,pls61,pls62,thesis]).

%% how many units for degree
degree_units(5).

%% hard precedences
after(pls50,pls60).
after(pls50,pls61).
after(pls51,pls62).

%% soft precedences
first([pls50]).
last([thesis]).
year(1,[pls50,pls51]).
year(2,[pls60,pls61,pls62]).
```

---

Table 4 shows a code extract showing how we flag illegal unit combinations. For example, the first code extract reads “a student who has already taken the modules listed in *AttendedBefore* may not take module *X* if module *PreReq*, which must precede *X*, is not listed therein”. Similarly, the second code extract reads “if module *X* must be taken last, then by the time the student has taken it, the overall number of units taken cannot be less than the number of units required for the degree”.

Table 4. Implementing hard and soft conflicts

---

```
hard_conflict(AttendedBefore,X):-
    after(PreReq,X),
    not(member(PreReq,AttendedBefore)).

soft_conflict(AttendedAfter,X):-
    last(U),
    member(X,U),
    length(AttendedAfter,L),
    degree_units(DU),
    L < DU.
```

---

Finally, Table 5 shows a code extract implementing topological ordering to generate all admissible tuition paths of four units (as required to proceed to a thesis). Essentially, it creates a graph where nodes are sets of selected units and edges are transitions between states. The output of the program can be edited and then visualized with *Graphviz*<sup>3</sup> or similar visualization software.

Table 5. Generating tuition paths

---

```
advance(X,X,[]).
```

---

<sup>3</sup> *Graphviz* is available at <http://www.graphviz.org>.

---

```
advance(Start,End,YearMoves):-  
    stepyear(Start,Next,Mv1),  
    advance(Next,End,Mv2),  
    append([Mv1],Mv2,YearMoves),  
    add_edge(Start,Next).
```

---

## 6. On the Validity of the Methodology and related System Issues

We have focused our design and implementation on providing us with initial coarse estimates for our postgraduate programme and on developing a proof of concept for the overall configuration. The latter is essential if we aspire to eventually build a system for organization wide adoption. Accordingly, we have identified three major technical issues for our research agenda.

The first issue has to do with ease of implementation. It is a variant of the problem that led us to develop the grammar described in Section 5 to minimize the verbosity of the original specification that would be daunting to check and verify. We expect the visual grammar to allow us to address a certain reservation towards formalisms that is sure to be exhibited among many of our colleagues. However, the final size of an Extend implementation and the repetitive nature of the Extend code are both conducive to allowing implementation errors to creep in. A key technical challenge is then to be able to directly generate Extend code excerpts and their interconnections by only using as input the visual grammar. In such a setting, we would still need to populate graph edges with transition probabilities but, even if we eventually opt to not enhance the grammar with probabilities, it would be far easier to fill such probabilities in predefined *blanks* in the generated code excerpts, especially so if we also decide to use default values for most of the transitions.

At a narrow tangent to this issue lies the option to reframe the requirements specification in another notation. The apparent similarity of the diagrams in Figure 1, Figure 2 and Figure 4 to Bayesian networks suggests that we might be able to employ more readily based solutions for modeling the available paths to graduation. While the eventual sidelining of the Extend system does not now seem as an option, the relatively recent adoption of belief networks as a tool for simulation modeling (Van Tol and AbouRizk, 2006) has also grasped our attention.

The second issue has to do with the credibility of the transition probabilities. Each academic year sports a different configuration of the student population, with new students arriving each year. A certain organizational memory develops gradually based on students' perceptions about which modules are best to select given one's time available for studying or which modules are easier to follow based on one's earlier enrolments. This memory is subject to change every year and the associated transition probabilities inevitably change. One may use the statistics of all previous years (implementing a time window), possibly discounting for distant years. Another reasonable option is to just use the statistics for the previous year. Whichever decision one makes, the question that looms is whether this is a decision that must be made at the programme or at the university level or, even more importantly, whether this is a decision that statistically speaking matters. It is fortunate that this second issue can be studied independently from the previous one using a theoretical or applied statistics toolkit and simulation.

The third issue has to do with the structural stability of any given degree programme and is the one with by far the most challenging consequences. As years go by, academic programmes change. When new modules are introduced or some modules are no longer offered, things are easy to model, because either the transition probabilities are calculated afresh or some transitions are trimmed. When, however, academic committees decide, and they rather often do, to move some modules up or down the academic requirements ladder, then modelling chaos occurs. While it is straightforward to deal with such issues in an administrative context, deciding how one changes the model and how transition probabilities are calculated for the new configuration based on the previous data now seems to be a daunting computational task. We have not yet investigated the options for dealing with this problem. At present, we believe that the best outcome would be if theoretical analysis and statistical simulation might offer us some insight as regards the range of such changes, up to which we might be able to use default values without having to worry about the inevitable errors that incur due to the (educational) system's acquired momentum.



We note that the resolution of the latter two issues may transcend the prototype nature of our current implementation and may require the adoption of specific model description languages or process algebras that facilitate reasoning about simulation and about model consistencies (Pooley, 2007), or the adoption of models that explicitly allow for fuzziness in the specification of probabilities (Gien and Jacqmart, 2005).

## 7. Conclusions and future directions

Modelling the size of a student population and how such a population is spread into modules of a programme with arbitrary precedence requirements among its modules is surely a statistical tool that can aid a programme's management to plan ahead in a proactive manner. In this paper we have shown the key technical ingredients of a system that can provide related estimates and we have also presented which aspects of such a system need further research, either in terms of system integration or in terms of robust modelling in circumstances of change.

We acknowledge that population modelling and estimation may be at a wide tangent to policies as practiced by today's universities. We also acknowledge that (even) the strategy consensus required for fielding such systems to actually support university administration may take indefinitely longer than the resolution of the technical and scientific issues that we have already identified. Actually, such consensus is probably of a political rather than a technological nature (Ringwood *et al.*, 2003).

Still, however, the emergence of modelling tools and methodologies like the one we have proposed in this paper, serves the need to equip educational managers with the insight they direly need in order to contemplate policy alternatives and to reflect upon past decision based on actual data (Scott, 2005). Even if the results never reach the level of statistical significance, it will always be up to innovative managers to use simulation as a tool for exploring what-if scenarios (and not as a fine-tuning tool) in order to reflect about future policy directions.

## 8. Acknowledgements

Michalis Xenos first implemented a short program to help steer junior undergraduate Informatics students at the Hellenic Open University through the course selection requirements and priorities.

Nicos Karacapilidis offered insight into the options available by industrial class simulation software and initiated the liaison between the authors.

## References

- Chang, G.-C., and M. Radi (2001). *Educational planning through computer simulation*. Paris, UNESCO.
- Gien, D., and S. Jacqmart (2005). Design and simulation of manufacturing systems facing imperfectly defined information. *Simulation Modelling Practice and Theory* 13: 465-485.
- Open Learning (2004), Special issue 19(1) on "Student retention in open and distance learning".
- Pooley, R. (2007). Behavioural equivalence is simulation modelling. *Simulation Modelling Practice and Theory* 15: 1-20.
- Ringwood, J.V., F. Devitt, S. Doherty, R. Farrell, B. Lawlor, S.C. McLoone, S.F. McLoone, A. Rogers, R. Villing and T. Ward (2005). A resource management tool for implementing strategic direction in an academic department. *Journal of Higher Education Policy and Management* 27(2): 273-283.
- Scott, D. (2005). Retention, completion and progression in tertiary education in New Zealand. *Journal of Higher Education Policy and Management* 27(1): 3-17.
- Van Tol, A.A., and S.M. AbouRizk (2006). Simulation modelling decision support through belief networks. *Simulation Modelling Practice and Theory* 14: 614-640.
- Webster, T. (1997). Cost analysis and its use in simulation of policy options: the Papua New Guinea education finance model. *International Review of Education* 43(1): 5-23.
- Xenos, M., Ch. Pierrakeas, and P. Pintelas (2002). A survey on student dropout rates and dropout causes concerning the students in the Course of Informatics of the Hellenic Open University. *Computers and Education* 39: 361-377.