An Assessment of Theory of Computation in Computer Science Curricula

C. M. Keet
University of KwaZulu-Natal, South Africa
keet@ukzn.ac.za

Computing curricula are regularly reassessed and updated to reflect changes in the discipline. Currently, the ACM/IEEE curriculum 'CS2013' is under review, which provides the main international guiding principles for curriculum development. We assess this curriculum focusing on one of the core themes of computer science, being Theory of Computation. We examine how it is implemented in computer science curricula around the world, and the sentiment around teaching it. Two surveys were conducted, examining curricula and syllabi of computer science degrees and an online opinion survey. Theory of Computation is part of 84% of the consulted curricula around the world, but taught at only 27% of the South African universities, and these syllabi contain substantially more Theory of Computation topics than the basic core in CS2013. The online survey not only confirms this but also indicates inclusion of even more topics as essential for Theory of Computation and shows that it is mostly solidly part of the degree programme as a core course and mostly in the 2nd or 3rd year, despite that for more than half of the respondents, the course causes issues in the university system.

Introduction

Computing skills are designated as scarce skills, yet it is important not only to produce more computer science graduates, but also of a high quality so that graduates are equipped with the capabilities to design novel IT solutions for the Southern African context. Currently, there is no South(ern) African computing organisation for quality and curriculum guidance and accreditation to assist with this, hence one is left with considering international efforts and adapt it to the regional context (Marshall, 2011). The main international organisations for Computer Science (CS), such as the US-oriented Association for Computing Machinery (ACM) and the International Federation for Information Processing (IFIP), conduct periodic updates to the CS curricula. A landmark publication in 1989 by the ACM defined the CS discipline and curriculum topics (Denning et al, 1989), which was followed by an internationally more inclusive and much more detailed list of topics and the notion of 'modular curriculum' by (UNESCO-IFIP, 1994), which was followed by the UNESCO-IFIP "ICF-2000" curriculum and "CC2001" by the ACM (Roberts, 2002). Currently, the ACM/IEEE "CS2013" is under public review (ACM/IEEE Joint Task Force on Computing Curricula, 2012; Sahimi, Aiken & Zalenski, 2010; Sahimi et al, 2012), which, as notable change to its predecessors, recognises the diversification within CS beyond the simple hardware/software/IT divides, and describes learning outcomes. A central theme is what falls under the banner of Theory of Computation (ToC)—being, roughly, formal languages, automata, Turing machines, computability, and complexity—that introduces the mathematical and computational principles that are the foundations of CS, such as the foundations of programming languages and algorithms, and the limits of computation; typical textbooks are Sipser (1997) and Hopcroft, Mottwani & Ullman (2007). These themes are mostly taught in at least one basic undergraduate course (module), but also can be split over two or more courses, such as automata in a compilers course and complexity jointly with algorithms. These topics were core in the original curriculum guidelines (Denning et al, 1989;

UNESCO-IFIP, 1994), but CC2001 and CS2013 have divided that into a smaller core and a set of elective topics, which contributes to a steady flow of anecdotes about 'dumbing down' of CS degrees (e.g., (Dijkstra, 1988; Dewar & Schonberg, 2008)). The ACM/IEEE and UNESCO-IFIP curriculum guidelines are, however, not to be understood as templates for course design, evaluation of subjects, teaching and learning principles, or criteria for accreditation, but principally as a guideline by means of a list of required and recommended undergraduate degree contents (van Veen, Mulder & Lemmen, 2004). Given the observed difference in emphasis on ToC in the international curricula guidelines over time and the pending update on the ACM/IEEE curriculum—as well as curriculum update discussions at the author's institution and elsewhere, e.g., (Sahimi, Aiken & Zalenski, 2010)—it is useful to assess 1) how ToC topics are implemented in CS curricula around the world at present, 2) whether there are any country or regional differences, and 3) what the sentiment is around teaching ToC in academia that may influence the former.

To gain insight in these three aspects, we conducted two surveys, which, to the best of our knowledge, are the first of its kind. The first survey examines syllabi of CS degrees as published online on the respective universities' websites with respect to ToC. The second survey was an online survey open to everybody, which asked for the respondents' opinions on ToC, the context in which it is taught, and what topics should be in a ToC course. The salient outcomes are that ToC is included in the vast majority of the consulted CS curricula around the world, except for South Africa, and these syllabi contain substantially more ToC topics than the basic core of CS2013. The online survey confirms this, and, moreover, the respondents generally include even more topics as 'essential' than that are given in the syllabi. Also, most respondents did do ToC and it is being taught at most of the universities with which the respondents are affiliated, and it is solidly part of the degree programme in the majority of responses. The opinion survey also highlights that there are difficulties in teaching ToC, exhibited by, among others, relatively high failure rates or other issues in the university system.

In the remainder of the paper, we first describe the materials and methods of the surveys, followed by the results and a discussion, and then we conclude.

Materials and Methods

The survey is divided into two components: the survey of extant curricula and syllabi and the online opinion survey. Limitations of the set-up will be addressed in the discussion section.

Curriculum and Syllabi Survey

General set-up. A selection of universities will be made, with a focus on all South African universities designated as 'traditional' or 'comprehensive', and from each major region in the world a subset of universities is pre-selected. The aggregations for the latter are: Africa outside South Africa, Asia, Latin America, Europe (continent, west), and the discontiguous region of Anglo-Saxon countries (Australia, Canada, New Zealand, UK, USA). The selection of the international universities is based roughly on inclusion in the Times Higher Education Ranking for countries unfamiliar to the authors. Of those regions and countries, only those will be assessed that have a website and a curriculum understandable in any of the languages that the author understands sufficiently in order to conduct the data collection (being, in alphabetical order: Afrikaans, Dutch, English, German, Italian, Portuguese, and Spanish).

Data collection. Parameters to record are: region, country, type of university (where applicable), whether ToC is included in a degree programme in whole or in part or not, and

if so in which year of the degree programme, and analysis of content of the syllabi (if present).

Opinion Survey

General set-up. The method chosen for this survey is that of an open online questionnaire for a two-week time period. Both individual email invitations are to be sent to colleagues inside and outside the University of KwaZulu-Natal, an announcement on the Description Logics and UKZN/CSIR-Meraka Centre for Artificial Intelligence Research mailing lists, and social media will be used as well (the author's Facebook and Google+ accounts). Respondents can leave their email to be contacted for further questions and feedback. The survey software used is LimeSurvey, because it is free survey software that has the necessary feature of conditional questions and extensive branching. Analysis of the results will be carried out with the built-in LimeSurvey features and Microsoft Excel.

Survey Questions. The survey consists of eight main questions, with conditional questions depending on the answer provided by the respondent. They are summarised here.

1. Should Theory of Computation [roughly: formal languages, automata, complexity] be a course in a CS programme?
    a. If yes: core or elective, undergraduate or postgraduate, when in the programme (undergrad (1, 2, 3), honours(4), MSc, PhD)?
2. Is it taught at your university?
    a. If no: was it, but cancelled? If yes: Why?
    b. If yes: core or elective, when in the programme (undergrad (1, 2, 3), honours(4), MSc, PhD), is it secure/solidly in the programme or threatened to be cancelled, does it cause 'problems' such as being flagged for low pass rates or negative student evaluations? Participation (indication): estimate of average amount of enrolled students in each course offering over the past 3 years (<10; 10-30; 31-60; 61-100; more than 100), is the amount stable or in decline (stable, slight decline, strong decline (>50% fewer students in past 5 years)). First-time pass rate (<20, 20-40, 41-60, 61-80, 80-100%).
3. Did you ever teach, or are currently teaching, Theory of Computation or Formal languages and automata only or Computability, complexity only?
4. Did you do Theory of Computation in your degree? If yes: which year of the programme? If no: do you miss it/regret not having done so?
5. Do you do research in Theory of Computation topics? Do you use Theory of Computation topics in research or at work?
6. Our survey of syllabi indicates some differences across universities. Please indicate for the following topics whether you consider the topic 'essential', suitable for an 'extended version' of a ToC course, or 'peripheral/may be skipped'. (For reasons of brevity, the full list of 46 topics in included in the results section only.)
7. Comments the respondent wishes to make, name of the organisation where the respondent works/studies, email (optional).

Results

In line with the two surveys, the results will be presented separately.

Computer Science Curriculum and Syllabi Survey Results

Countries and Universities. The full list of universities consulted is included in the appendix; they are 17 universities from South Africa, 15 from Europe, 15 from the Anglo-Saxon countries, 6 from Asia, 7 from Africa other than South Africa, and 8 from Latin America. The lower amounts for Africa, Asia, and Latin America is largely due to the language barrier and that they generally have less information online, except for the relatively famous (and rated) ones. A total of 9 universities had no or insufficient data online to assess ToC in the curriculum, being two in South Africa (Limpopo and Venda) and 7 in Asia, Africa, and Latin America.

ToC in the curriculum. The inclusion of ToC in part or in whole in the CS curricula of the South African universities that had such information online (n=15) with that of the consulted universities in the rest of the world that had such information online (n=44) is compared in Figure 1. More precisely: 27% of the South African universities—University of KwaZulu-Natal, University of South Africa, University of Western Cape, University of Witwatersrand—has ToC in the CS curriculum compared to 84% elsewhere in the world.

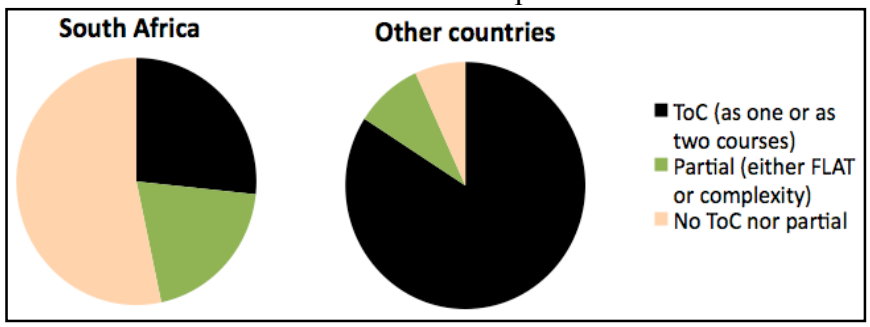

Figure 1. Comparison of ToC in the CS curriculum in South Africa and in other countries. (FLAT=formal languages and automata theory)

Disaggregating the "other countries" by the identified regions and including also those universities consulted for which no or insufficient material was available online, we see a similar picture (see Figure 2). The African universities consulted that include ToC in the curriculum are Alexandria, Makerere, Zimbabwe, Lagos, and Kenyatta partially, whereas for Botswana, and Addis Ababa no information was available online. The absence of online information was also an issue for the Latin American universities, but, given their responses on the opinion survey (see below), it definitely will amount to a higher percentage than is currently included in the syllabi data. This is unclear for Asia.

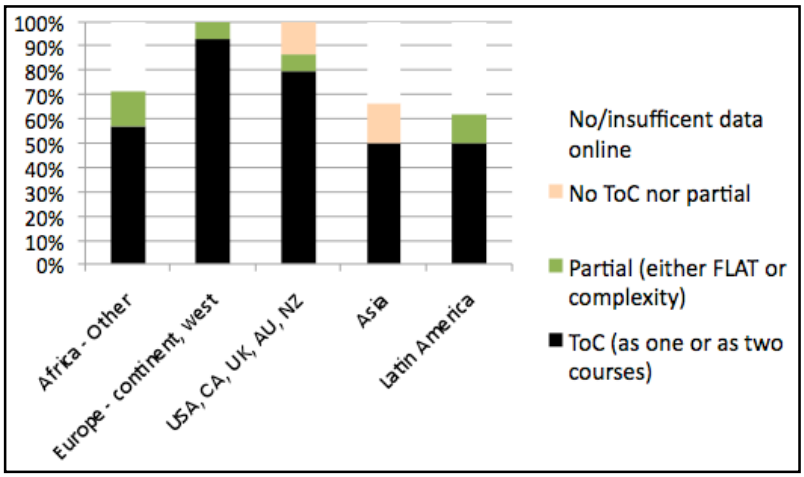

Figure 2. Curriculum evaluation on inclusion of ToC, disaggregated by region outside South Africa.

The difference between Europe (93% inclusion of ToC) and the Anglo-Saxon countries (80%) may be an artefact of the sample size, but it deserves further analysis with larger sample sizes.

The level of consistency of presenting information online about curricula and/or course listings and/or course descriptions and/or detailed syllabi varied widely, and in several cases multiple options applied. For instance, ToC being core in one 'track' or 'stream', but not another, or for some students scheduled in the BSc and others in the MSc programme. There were 43 universities (of the 59 with data) that had sufficient information online regarding timing of ToC in the degree programme. Of those, only 5 had it explicitly in the MSc degree programme, with the rest mainly in year 2, 3, or 4. Five universities that offer ToC have it spread over 2 or more courses, and the rest offers it in a single course (occasionally, additional advanced courses covering advanced topics such as probabilistic automata and other complexity classes). An initial attempt was made to categorise the topics of the syllabi in more detail, but this was abandoned due to the high variability of detail of such information being online.

Opinion Survey Results

Characterisation of respondents. The number of completed surveys was 77, of which 58 filled in their affiliation. The respondents are mainly employed at universities and research institutes: 12 respondents indicated an academic affiliation in South Africa (Stellenbosch, KwaZulu-Natal, UNISA, Pretoria, Witwatersrand) and thus the majority of respondents were from around the world, including universities of Illinois, Rutgers [US], Waterloo [CA], Bolzano [IT], Dresden [DE], Geneva [CH], Southampton [UK], Ben Gurion [IL], Bahía Blanca [AR], CENATAV [CU], Pontificia [CL/BR], São Paolo, Rio de Janeiro, Sergipe, Espirito Santo, and 8 others from [BR], Macau [CN] and Indonesia [ID], and at least three respondents are employed in industry (Boeing, Google, and SRI International). 35 respondents indicated to have taught ToC, 37 formal languages and automata only, 25 computability and complexity, and 16 closely related courses such as algorithms, logic, and compilers. 41% of the respondents (32) conducts research in ToC topics and 72% (56) use it in research or other work (including 2 from the aforementioned companies). Twelve respondents neither conduct research in ToC topics nor use it. The spread of respondents is too broad to merit statistical analyses about whether responses vary significantly by country, by continent, or by predominant language at the organization.

Answers relating to ToC. The anonymised raw question answers with percentages, sorted by question (exported from LimeSurvey) are online at http://www.meteck.org/files/tocsurvey/, and anonymised answers to each survey question in per-question-format with answers by respondent as a spreadsheet are available upon request; the following paragraphs summarise the results.

The basic ToC course statistics are as follows. 76 of the 77 respondents are of the opinion ToC should be in programme, 74 (96% of the answers) have it currently in the programme, and 82% had it in their degree programme when they were a student. Of those 14 who did not do ToC during their studies, 86% misses not having had that opportunity and the remaining respondent did not do ToC, does not miss it not having done so, said that it should not be in the programme, and never uses it at work.

Considering when it should be taught in the degree programme, and taking note of the 10 comments in the comments field of the survey describing it has been divided over several courses, there was a clear tendency for undergraduate compared to honours, MSc, or PhD (ratio yes/no 0.69 for undergrad versus 0.45 for honours and 0.33 for MSc/PhD), and the 3rd year in particular (ratio yes/no year1 0.13, year2 1.03, and year3 1.57). This roughly

corresponds to when the respondents themselves did ToC and when it is taught at the university. Given there must be several calendar years difference between being a student and that the survey respondents are graduated since a while (see previous paragraph), there is thus little change over the years, as can be seen in Figure 3. The 5% increase in teaching ToC in year 3 comes mainly from the decrease in honours/year 4 and PhD.

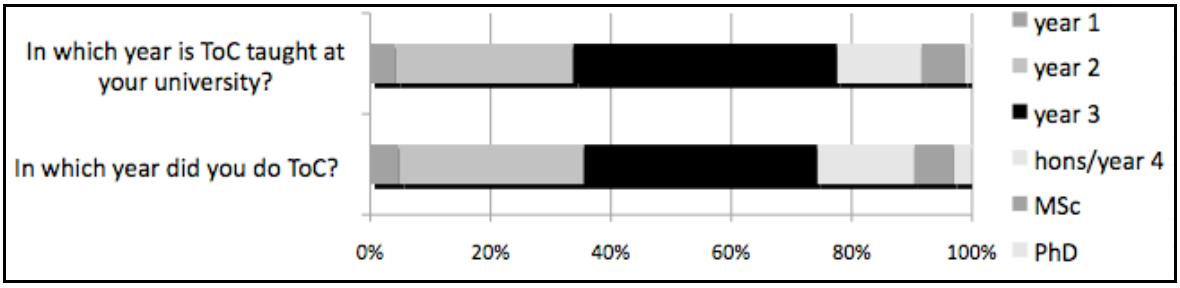

Figure 3. Comparison between when the survey respondents did ToC in their studies and in which year in the study programme it is taught at their university (where applicable).

ToC is a core course in the curriculum in 67 of the 74 answers (90%), and secure in the programme for 57 out of 66 responses (86%). Only few reasons were provided for "under threat/other", being that it has been removed from some specialisations but not all (in part, due to the computer science vs. information systems tensions, as elaborated on by one respondent), or threatened due to low enrolment numbers, resulting in one case ToC being taught only every other year. Enrolment numbers vary greatly, from less than 10 students (6% of respondents) to classes with more than 100 students (16%), with the main 31-60 students/year (40%) and then 11-30 students (24%). 25% (15 of 60) record a slight decline in enrolment.

Given the plentiful anecdotes, hearsay, and assertions in other ToC teaching papers about difficulties with ToC teaching and learning, we also asked about that in the survey. The data provided by the respondents do substantiate the existence of issues to some extent. While 44% of the respondents answered that there are no issues and everything runs smoothly, 32% note it causes problems in the academic system each year and another 24% reported that management/student affairs has gotten used to the fact there are problems, i.e., a slight majority of respondents faces issues. Several respondents provided additional information regarding the issues, mentioning low pass rates (3), that students struggle because they do not see the usefulness of ToC for their career (4), that it also depends on the quality of the teacher (2), and low enrolment numbers (2). We considered three variables present in the result set that may influence there being issues: pass rates, class size, and content of the course. Course content was extrapolated from the answers given to the ToC topics, where more topics denoted as 'essential' was assumed to result in a heavier course load, which need not be the case, and no correlation was found. Data on the interaction between pass rates, participation, and issues were analysed: for 45%, the first-time pass rates remain below 60% and with 80% of the respondents, the pass rate remains below 80%. There is no clear trend between pass rate, class size and issues, except that pass rate 41-60% and class size 31-60 have comparatively more issues (72% and 64% of the reported instances, respectively), and that the correlation between pass rate and issues is 0.79; *n* is to small to draw any conclusions for the other combinations.

ToC topics. The final set of main questions concerned the topics that should be part of a ToC course. 46 topics were listed and for each one, the answer [essential/extended/ peripheral/no answer] could be given. The responses were analysed in two different ways: calculating the percentage of a response value out of the total responses given and by

assigning values to the responses (essential = 3, extended = 2, peripheral = 1) and ordering topics according to the average over the given answers. The order of the topics is roughly the same for the essential topics and varies only by a few places at the tail end; thus, there is a consensus about which topics are important regardless the measure. The complete list of ToC topics ordered on percent 'essential' is shown in Table 1.

Table 1. Ordering of the 46 ToC topics, by calculating the percentage of responses that marked it as 'essential' out of the given answers.

Topics ordered on percent 'essential'		***cont'd***	
Regular expressions	98.55	Show problem to be decidable/un-decidable(RE, non-RE, RE but not Rec.)	61.90
Deterministic Finite Automata (DFA)	95.71	Push-down Automata (PDA,deter-ministic and non-deterministic)	61.54
The Turing Machine (basics)	94.37	Converting RG NFAs	60.61
Context-free Grammars (defini-tion, ambiguity, simplification, derivations)	94.20	TM acceptors	57.14
Non-Deterministic Finite Automata (NFA, epsilon-NFA)	88.41	Pumping lemma for CFLs	54.69
Equivalences & conversion RE and automata	85.29	Equivalences & conversion PDA, CFG	51.61
Problems a computer cannot solve	85.29	Show problem to be P, NP, NP-complete, NP-hard, co-NP	50.75
Halting problem	82.81	State minimization	49.25
Properties of Regular languages	80.30	Closure properties of CFLs	49.21
Regular grammars (RG)	80.00	Recursively Enumerable, non-RE, RE but not recursive languages	46.38
Examples of some undecidable problems/languages	78.13	Decision properties of CFLs	46.03
Church-Turing thesis	77.94	Non-deterministic TM	45.45
Computability and decidability	76.81	Cook's theorem	38.71
Equivalences & conversion DFA, NFA, epsilon-NFA	73.85	Diagonalization language	34.38
P, NP, NP-complete, NP-hard, co-NP	73.53	Multi-tape TM	30.88
Universal Turing Machine	72.06	Rice's theorem	30.00
Undecidability	68.57	Post correspondence problem	26.67
Pumping lemma for Regular languages	68.18	PSPACE, EXPTIME, ...	26.23
Some well-known NP problems (e.g., TSP, SAT, Node cover)	68.18	Moore Machines	21.31
Chomsky normal form, hierarchy	67.16	Mealy Machines	21.31
Reductions	65.15	Programming tricks for TM (storage, tracks)	20.59
Proving undecidability	62.86	TM transducers	17.74
Polynomial time reductions	62.69	Hot/fun topics and their complexity classes (e.g., games)	15.87

Topics that received most (50-60%) 'In an extended course' are: TM transducers; Post correspondence problem; PSPACE, EXPTIME, ...; hot/fun topics and their complexity classes (e.g., games); 'programming tricks' for TM (storage, tracks); and Cook's theorem. Four respondents used the comments field to add other topics, being: weighted automata and transducers with applications in a basic course and tree automata in an extended course, Quantum Turing Machine, Savitch Theorem, PSPACE-completeness, and parallel complexity classes with PRAM NC P-complete.

Discussion

The responses of the opinion survey—77 being a substantial amount for an open survey—show an overwhelming agreement about the need for a ToC course in a CS degree programme, regardless whether they have done the course themselves, teach or have taught it or a similar course, or conduct research in it, or do not us it at all. As such, it re-confirms ToC's place in the curriculum from the perspective of, mostly, academics.

Concerning topics of a ToC course, it is perceived decidedly that formal languages, automata, Turing machines, complexity, computability and decidability themes form part of one coherent offering, but that the detail of the sub-topics covered may vary. For instance, including Turing machines in a basic ToC course, but transducers, storage and tracks only in an extended or advanced ToC course, including Deterministic and Non-Deterministic Finite Automata, but not Mealy and Moore machines, and covering Context-Free Grammars, but not decision and closure properties. This contrasts quite markedly with the Strawman/CS2013 outline, which is depicted in Figure 4.

AL/Basic Automata Computability and Complexity
[3 Core-Tier1 hours, 3 Core-Tier2 hours]
Topics:
[Core-Tier1] Finite-state machines, Regular expressions, The halting problem
[Core-Tier2] Context-free grammars (cross-reference PL/Syntax Analysis), P vs. NP (tractable and intractable problems), Definition of P, NP, and NP-complete, Exemplary NP-complete problems (e.g., SAT, Knapsack)

AL/Advanced Computational Complexity
[Elective]
Topics: Review definitions of the classes P and NP; introduce EXP, NP-completeness (Cook's theorem), Classic NP-complete problems, Reduction Techniques

AL/Advanced Automata Theory and Computability
[Elective]
Topics: Sets and languages, Regular languages (Review of deterministic finite automata (DFAs), Nondeterministic finite automata (NFAs), Equivalence of DFAs and NFAs, Review of regular expressions; their equivalence to finite automata, Closure properties, Proving languages non-regular via the pumping lemma or alternative means), Context-free languages (Push-down automata (PDAs), Relationship of PDAs and context-free grammars, Properties of context-free languages), Turing machines, or an equivalent formal model of universal computation, Nondeterministic Turing machines, Chomsky hierarchy, The Church-Turing thesis, Computability, Rice's Theorem, Examples of uncomputable functions, Implications of uncomputability

Figure 4. Proposed CS2013's ToC topics in the Strawman draft (layout edited).

However, the feasibility of imparting a real understanding of complexity classes P and NP without also touching upon computability and Turing machines is limited. In addition, the hours indicated in Figure 4 have to be understood as minimum hours of fact-to-face

lectures, which amounts to 8 lessons at a South African university (8 * 45 mins = 6 * 60 mins), or at least almost 3 weeks of a standard 16 credit semester course, which, if this minimum is adhered to, amounts to a *very* superficial treatment of partial ToC topics. We focus explicitly on CS programmes, which should have most time dedicated to ToC compared to other computing specialisations, but even then, the voices from the field clearly demonstrate putting a higher weight on ToC than the ACM/IEEE curriculum developers allot to it. Why could this be so? Arguments can be heard that ToC matters more for CS than other recently recognised specialisations within computing—e.g., software engineering, net-centric computing, information systems, and computational biology—, that this diversification has to be recognised by the curriculum developers (Rosenbloom, 2004; Sahimi et al, 2012), and that it should result in putting more or less weight on the core topics (see (ACM/IEEE Joint Task Force on Computing Curricula, 2005) for a detailed analysis on sub-disciplines within computing and a proposed weighting of curriculum themes). This is not reflected as such in the Strawman draft: the different tracks *within* CS all have to do the core, with 100% for tier-1 and >80% of tier-2 core, and an undefined amount of the elective topics to facilitate track-development, although no tracks have been defined in the CS2013 yet. It is noticeable only when one compares it with the Software Engineering curriculum guidelines (ACM/IEEE Joint Task Force on Computing Curricula, 2004), as those guidelines include only a little bit on finite state machines, grammars, and complexity and computability in the "Data structures and algorithms" and "Discrete structures II" themes. It may be the case that, in praxis, those degree programmes called "computer science" indeed do contain the more fundamental topics, such as ToC (and logic, formal methods etc.), and that other 'tracks' actually have been given different names already, hence, would have been filtered out unintentionally *a priori* in the data collection stage of the curriculum survey.

Concerning issues teaching ToC, on an absolute scale, that 56% faces issues with their ToC courses is substantial. It deserves a comparative analysis to uncover what the other half does so as to not have such issues. Comments in the survey and offline (in the form of follow-up emails) by survey respondents suggest it may help to demonstrate better the applicability of ToC topics in the students' prospective career, have experienced good teachers, and appropriate preparation in prior courses to increase the pass rates. Further, it might be related to the quantity and depth of material covered in a ToC course with respect to nominal course load. The data hints also to another possible explanation: even with a 80-100% pass rate and no low enrolment the 'gotten used to the issues' was selected occasionally, and vv., with a 41-60% pass rate that everything runs smoothly, thereby indicating that having issues might also be relative to a particular university culture and expectations of students, academics, and management.

Addressing possible limitations of the survey set-up. There are several limitations of the set up of the surveys that may affect the results. First, the non-South African universities were selected largely based on reputation, such as the Times Higher Education Ranking, to the extent one might ponder whether the fact that it is skewed toward the 'good' universities may have an effect, and one might argue that they include ToC because they have a good insight in designing a high quality curriculum. Or, that the sample is too focussed on education systems in 'the West'—which holds for the syllabus survey, but not the opinion survey—and that such curricula are, or have to be, adapted to the local context in a yet to be specified way. The latter ought not to entail omitting core material from a discipline's curriculum anyway, and, moreover, CS and development of novel and good quality software requires an understanding of ToC topics; e.g., in order to develop a correct isiZulu grammar checker or parser, scalable image pattern recognition algorithms to

monitor wildlife tracks with pictures taken *in situ*, or an ontology-driven user interface for the South African Department of Science & Technology's National Recordal System for indigenous knowledge management. These are broad-sweeping statements and this research does not provide, and did not aim to provide, an answer to these questions. Second, when there were detailed syllabi, they often did not provide detail regarding the hours spent on each topic and no credit comparison was made, which can introduce a larger variability than what is presented in the results. A case in point is the detailed evaluation of the curricula by academics from Imperial College London, TU Delft, ETH Zurich, and RWTH Aachen that gave ToC a relative importance of 10%, 7%, 4.5%, and 9.3%, respectively (IDEA League, 2001). However, this does not change the substantial difference between the presence/absence of ToC in the curricula in South Africa versus in other countries.

Regarding the opinion survey, the distribution of the invitations was skewed toward fellow scientists who work in similar fields, both regarding individual emails and the distribution lists, most of whom use, or know they benefit from, ToC topics in their work, and perhaps a bit skewed toward computability and decidability as most important topics. Also, it is an open survey, hence, it may exhibit the tendency that those who are more passionate about ToC will be the main proportion of respondents. However, the cross-check with the curricula and syllabi survey did not demonstrate a clear difference, and the necessity of including at least complexity is also reflected in the Strawman/CS2013 draft; hence, the opinion survey does not indicate the existence of this possible bias. Moreover, the respondents come from each continent of the world and from many different universities, therewith providing a valuable snapshot of various aspects of ToC, in particular regarding the basic ToC course statistics, perceived importance of topics, and data about prevalence of issues, which is, to the best of our knowledge, the first of its kind.

Returning now to the original three questions posed in the introduction, we can answer them as follows. Regarding the first question, on how ToC is implemented, it can be observed that ToC topics in the actual international curricula are more in line with the older curriculum guidelines of Denning et al (1989) and UNESCO-IFIP (1994) than the more recent versions that put less weight on ToC topics. The timing in the curriculum regarding when to teach ToC remains largely stable. The results show there are country/regional differences, with the most pronounced one being that ToC is taught at only 27% of the South African traditional and comprehensive universities versus its inclusion in 84% of the consulted curricula elsewhere in the world. Even including the ones with partial ToC coverage does not make up for the differences with elsewhere in the world and any of the proposed curriculum guidelines. Other geographic or language-based differences are not deducible from the data, or: based on the data, region does not matter substantially regarding inclusion of ToC in the CS curriculum, except that the slight difference between Europe and the Anglo-Saxon countries deserves further attention. Opinion on ToC is overwhelmingly in favour of having it in the curriculum, and primarily in the 2nd or 3rd year. Also, a large list of topics is considered to be 'essential' to the course, and this list is more inclusive than the recent international curricula Strawman drafts' core for ToC topics. Despite noted issues with the course, the voices from the field clearly indicate that ToC is here to stay.

Conclusions

Both the survey of the international curricula and syllabi and the opinion survey show an overwhelming agreement that Theory of Computation should be taught and is being taught, and a majority has it scheduled in the 2nd or 3rd year in an undergraduate computer

science programme. The course is mostly solidly in the programme as a core course. There is agreement on the typical topics that are considered as essential to Theory of Computation, covering regular and context-free languages, automata, Turing machines, undecidability, computability and complexity, where the subtopics covered vary. This is in line with older computing curricula guidelines, but less so with recent proposals that, comparatively, downplay Theory of Computation topics even for pure/'majoring in' computer science curricula in favour of a smaller core and multiple tracks. About half of the respondents note there are issues with the course, for various reasons, including, but not limited to, low pass rates and low enrolment, where roughly half observe first-time pass rates below 60%. This nevertheless does not to have an effect on the curricula thus far.

Given that, practically, Theory of Computation is solidly in the CS degree programme, and perhaps ought to be introduced more widely in South Africa, our future line of work pertains to assessing reasons behind the noted issues with teaching Theory of Computation, and what makes the course successful and running smoothly in some cases, including aiming to obtain a better specification of its prerequisites, and therewith other undergraduate courses.

References

ACM/IEEE Joint Task Force on Computing Curricula. (2012). *Computer Science Curricula 2013 Strawman Draft (Feb. 2012)*. ACM/IEEE, Online: http://ai.stanford.edu/users/sahami/CS2013/.

ACM/IEEE Joint Task Force on Computing Curricula. (2005). *The overview report*. ACM, AIS, IEEE-CS, September 30, 2005.

ACM/IEEE Joint Task Force on Computing Curricula. (2004). *Software Engineering 2004*. ACM, IEEE-CS, August 23, 2004.

Denning, P.J., Comer, D.E., Gries, D., Mulder, M.C., Tucker, A., Turner, A.J. & Young, P.R. (1989) Computing as a discipline. *Communications of the ACM*, 32(1), 9-23.

Dewar, R.B.K. & Schonberg, E. (2008). Computer Science Education: where are the software engineers of tomorrow? *Crosstalk: The Journal of Defense Software Engineering*, January 2008, 28-30.

Dijkstra, E.W. (1988). On the cruelty of really teaching computing science. Unpublished manuscript EWD 1036.

Hopcroft, J.E., Motwani, R. & Ullman, J.D. (2007). *Introduction to Automata Theory, Languages, and Computation*, 3rd ed., Pearson Education.

IDEA League (2001). *Report on comparison of curricula in Computer Science*. ICL, TUD, ETH & RWTH. Oct, 12, 2001. Online: http://www.automata.rwth-aachen.de/~thomas/Ideareport01.pdf.

Marshall, L. (2011). Developing a Computer Science curriculum in the South African context. In: G. van der Veer, P. Sloep & M. van Eekelen (Eds.), *Computer Science Education Research Conference* (pp9-19). Heerlen, the Netherlands, 7-8 April, 2011. New York: ACM Conference Proceedings.

Roberts, E. (2001). *Computing Curricula 2001: Computer Science Final Report*, New York : IEEE Computer Society.

Rosenbloom, P.S. (2004). A new framework for computer science and engineering. *IEEE Computer*, 37(11), 23-28.

Sahimi, M., Aiken, A, & Zalenski, J. (2010). Expanding the frontiers of computer science: designing a curriculum to reflect a diverse field. In: G. Lewandowski, S. Wolfman, T.J. Cortina & E.L. Walker (Eds.), *Proceedings of the 41st ACM technical Symposium on Computer Science Education* (pp. 47-51). Milwaukee, USA, March 10-13, 2010. New York: ACM Conference Proceedings.

Sahimi, M., Roach, S., Cuadros-Vargas, E. & Reed, D. (2012). Computer Science curriculum 2013: reviewing the Strawman report from the ACM/IEEE Task Team. In: *Proceedings of the*

43rd ACM technical Symposium on Computer Science Education (pp. 3-4). Raleigh, North Carolina, USA, February 29 - March 3, 2012. New York: ACM Conference Proceedings.

Sipser, M., *Introduction to the Theory of Computation*, PWS Publishing Company, 1997.

UNESCO-IFIP. (1994). *A modular curriculum in computer science*. UNESCO and IFIP report ED/94/WS/13. 112p.

van Veen, M., Mulder, F., Lemmen, K. (2004). What is lacking in curriculum schemes for Computing/Informatics? In: R. Boyle, M. Clark, & A. Kumar (Eds.), *Proceedings of the 9th Annual SIGCSE conference on Innovation and Technology in Computer Science Education* (pp186-190). Leeds, UK, June 28-30. New York: ACM Conference Proceedings.

Acknowledgements

We are grateful for all the people who took the time to fill in the survey, and especially those who provided additional offline/email feedback and information. We also would like to thank Leonard Els for setting up the survey software.

Appendix A: Universities consulted for the syllabi survey

South African universities (17): UCT, UFH, UFS, UKZN, Limpopo, NWU, UP, RU, Stellenbosch, UWC, Wits, UJ, NMMU, Unisa, Univen, WSU, UniZulu.

European (continent, west) universities (15): Free University of Bozen-Bolzano and "la Sapienza" University of Rome [IT], Technical University of Dresden, Technical University of Rhine-Westphalia in Aachen (RWTH Aachen), and Technical University of Munich [DE], Free University of Amsterdam and Technical University of Eindhoven [NL], Catholic University Louvain and Ghent University [BE], Vienna University of Technology [AT], Linköping University [SE], ETH Zurich and EPFL [CH], Polytechnic University of Madrid and Polytechnic University of Catalonia [ES].

Universities in Anglo-Saxon countries (15): Oxford University, Manchester, and University of Edinburgh [UK], Toronto and Vancouver UBC [CA], MIT, Stanford, Penn State, Yale, Harvard, Stony Brook, and CMU [USA], University of Melbourne and Australian National University [AU], and the University of Auckland [NZ].

Asian universities (6): University of Malaya [MY], National University of Singapore, Peking university and the University of Hong Kong [CN], University of Tokyo [JP], and the Indian Institute of Technology, Bombay.

African (non-SA) universities (7): Alexandria University [EG], Makerere [UG], University of Zimbabwe, University of Lagos [NG], Kenyatta University [KE], University of Botswana, and Addis Ababa University [ET].

Latin American universities (8): UCI and University of Havana [CU], Bolivarian University of Venezuela, Pontifical University of Chile and University of Chile, State University of Campinas and University of São Paolo [BR], and National Autonomous University of Mexico.