Origins and Metamorphoses of The Trinity: Logic, Nets, Automata

Boris Trakhtenbrot
School of Mathematical Sciences
Tel Aviv University
Ramat Aviv 69978, Israel
e-mail: trakhte@math.tau.ac.il

Synthesis and verification of systems with finite state space are well established problems in Logic and Computer Science and have a long history. Precise formulations are based on preliminary formalization of two languages: SPEC (for specifications), IMP (for implementations) and a (satisfaction) relation *sat* included in IMP $\times$ SPEC.

SYNTHESIS: Given s in SPEC. (i) Does *imp* exist in IMP such that *imp sat s* holds? (ii) If yes - construct such an *imp* (all of them?!).

The problem can be relaxed to VERIFICATION: Does *imp sat s* hold for given *imp* and s?

The solutions sought are algorithms which provide the correct answers and/or constructions.

Of course there is also the problem of EFFICIENCY: Estimate and improve the complexity of the algorithms and/or the succinctness of the results they provide.

Clearly, one can consider different languages which may reflect a variety of abstraction levels. It may well happen that an object at a given level may serve as implementation for a higher level and also as specification for a lower level. From this perspective the three-level paradigm is instructive and there is a proliferation of its versions. Though many of them are relevant to the subject, in this lecture I single out only one to which I refer to as The Trinity, namely:

1. At the highest level - specifications expressed as formulas based on Second Order Monadic Logic (SOML).
2. At the intermediate level - formalization of transducers (i.e. transformers of input signals into output signals) via finite sequential automata.
3. At the lower level - formalization of discrete synchronous hardware via logical nets.

Correspondingly we refer to behavioral synthesis (from 1 to 2) and to structural synthesis (from 2 to 3). Similarly for verification, etc...

The years 1956-61 marked a turning point in the behavioral track of the problems, as reported by Church at the 1962-International Mathematical Congress under the title "Logic, Arithmetic, Automata" (yet, another trinity!). Here is a quotation from there: "This is a summary of recent work in the application of mathematical logic to finite automata, and especially of mathematical logic beyond the propositional calculus".

Note that logical nets are not mentioned in the title though they figure (even if marginally) in Church's lecture; so, the focus there is on the "behavioral" aspects of the problems.

Church's lecture provides a meticulous chronology of events (dated when possible up to months) and a benevolent comparison of his and his student J. Friedman's results with work done by Buchi, Elgot and Trakhtenbrot. Nevertheless, in the surveyed period (1957-62) the run of events was at times too fast and thus omission prone. That is why his conclusion: "all overlaps to

1043-6871/95 $4.00 © 1995 IEEE

some extent, though more in point of view and method than in specific content" needs some re-examination.

The first (historical) part of my lecture is aimed at compensating for these circumstances and also introduces the structural aspects of the problems into the picture. In this way I hope to give a more accurate account of the origins of The Trinity and of the ulterior development which culminated with the full solution for behavioral synthesis by Buchi and Landweber (using the game theoretic approach suggested by McNaughton).

The second part of the lecture tackles the question: how and to what extent did The Trinity anticipate and promote developments in Logic and Computer Science?

Here the focus is on three points. The first two concern The Trinity as originally tailored for discrete-asynchronous systems:

(i) Logical nets as precursors of more elaborate models for concurrency implementations.

(ii) (Linear) Temporal Logic as a successful sugaring for SOML.

More recent developments in the theory of real-time systems suggest the third one:

(iii) Lifting the "classical" trinity to real-time systems.

Some results on timed automata point to the possibility of such a lifting with regard to automata and logic, but there is more to be done for logical nets in the context of real time.