Formulation of SOC Test Scheduling as a Network Transportation Problem

Sandeep Koranne*
Tanner Research Inc., Pasadena, CA

Vishal Suhas Choudhary
Philips Research Labs, Eindhoven

Reusability of tests is crucial for reducing total design time. This raises the problem of test knowledge transfer, physical test application and test scheduling. We present a formulation of the embedded core-based system-on-chip (SOC) test scheduling problem (ECTSP) as a network transportation problem. The problem is $\mathcal{NP}$-hard and we present a $O(mn(m+2n))$ 2-approximation algorithm using the result of the single source unsplittable flow problem. We describe the single source unsplittable flow problem (UFP) as given in [1]; let $G = (V, E)$ be a capacitated directed graph with edge capacities $c : E \rightarrow \mathbb{R}^+$, a source s and k commodities with terminals t_i and demands $d_i \in \mathbb{R}^+$, $1 \leq i \leq k$. A vertex may contain a number of terminals. For each i, we would like to route d_i units of commodity i along a single path from s to the corresponding terminal so that the total flow through an edge e is at most its capacity $c(e)$.

Formulation of ECTSP as UFP: Let us assume w.l.o.g that there are n tests for the n cores in a system. Each of these n tests has a testability requirement of b_i bits to be transported to and from the core, $1 \leq i \leq n$. Let there be m test resources like TAMs of width $w_1 \leq w_2 \leq \ldots \leq w_m$ bits respectively and BIST resources. We define $\gamma_{ij} = \frac{w_i}{w_j}$.

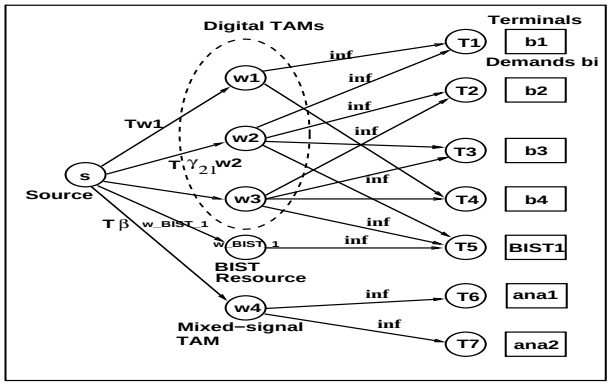

Figure 1. Formulating ECTSP as UFP

We now construct a graph $G = (m \cup n \cup s, E)$, where m is the set of test resources, n is the set of tests and s is a source vertex. Each test vertex i has a terminal which has an associated demand b_i of a commodity (the commodities are test bits). The edge set E is defined as follows; for each test $t_i \in n$ which can be executed with test resource m_j we add an edge between vertex i and j. The capacity of this edge e_{ij} is ∞. We add an edge e_{sj} between s and each test resource j of capacity $T\gamma_{j1}w_j$ for TAMs and $T\beta w_{BIST_j}$ for BIST resources.

Implementation: We have implemented a Test Planner Tool TFLOW with the Common Lisp language using the UFP algorithm of [1]. We compared the makespan length with ILP methods; the results are given in Table 1, executed on a Pentium IV machine with 128 MB of RAM. The table entries contain the makespan schedule in clock cycles. We improve upon the results of [2] not only in term of solution quality but significantly in computational time. The polynomial behaviour of our algorithm is clearly superior to the ILP methods employed recently.

Table 1. Comparative schedules of SOC $d695$

	Results of [2]		Our Experimental Results TFLOW		
	lpsolve		Schedule		Partition
W	$R\|\|C_{\max}$ Sch. Σ	Time (min)	UFP Sch. Σ	Time (sec)	π
16	42476	16.7	43290	0.0	(9,5,2)
20	34493	26.0	35502	0.0	(17,2,1)
24	28862	45.8	29023	0.0	(18,5,1)
28	26964	57.6	24701	0.3	(16,5,5,2)
32	22945	78.6	21567	0.2	(17,9,6)
36	19858	118.0	19757	8.4	(10,10,7,5,4)
40	19573	163.6	18799	13.9	(12,12,7,5,4)
48	18999	$180^{\dagger}$	17210	3.3	(19,19,6,4)
52	16928	$180^{\dagger}$	16084	3.9	(32,8,6,6)
56	15694	$180^{\dagger}$	13488	3.4	(20,16,16,4)
60	15694	$180^{\dagger}$	12611	3.5	(19,19,18,4)
64	15694	$180^{\dagger}$	12466	3.3	(20,20,20,4)

References

[1] Y. Dinitz, N. Garg, and M. X. Goemans. "On the single source unsplittable flow problem". *Combinatorica*, 19(1):17–42, 1999.

[2] V. Iyengar and K. Chakrabarty. "Co-optimization of Test Wrapper and Test Access Architecture for Embedded Cores". In *Proc. IEEE European Test Workshop (ETW)*, pages 189–194, May 2001.

*This work was performed when the author was with ED&T/Test, Philips Research. Email: sandeep.koranne@tanner.com.