Some Investigations into Parasitic Computational Approaches in Timetabling

George M. White

School of Information Technology and Engineering,
University of Ottawa,
Ottawa K1N 6N5, Canada
white@site.uottawa.ca

Abstract

Within the past decade there has been a remarkable interest in many new methods of constructing timetables, schedules and rosters [Bar95] . This has been aided by an interest in applying new techniques from artificial intellegence to more traditional operational research methods. More impetus has been given to this area by the realization that successful application of these new techniques can result in real savings in both money, time and personnel.

It has been observed that the application of these techniques to real problems requires a certain observable time. This is quite acceptable for many applications but may render certain problem instances unacceptably long for applications with real-time constraints. Even moderately large problems can require run times that may be unacceptably long [WXZ04]. The scalability of many heuristic methods is such that they are unsuitable for very large problems.

One way of approaching these large problems is through real parallelism, *i.e.* by using more than one physical scheduling engine.

- super-computers: Networks of specialized, homogeneous computers are programmed using supersets of well known computer languages.
- regular computers: Networks of PCs are connected together via Ethernet connections and specific software. The most common type of such networks is known as a *Beowulf cluster.* [1]
- off line computers: Massive numbers of off line computers work on slight variations of a common problem. *e.g.* the SETI project. [2]
- biological organisms: DNA strands are used to hold the solutions of problems by a suitable coding [LWF$^+$00].
- parasitic computing: many unwitting computers are used to solve problems.

This last method was introduced by Barabási *et al.* [BFJB01] by using computers in many parts of the world via the world wide web in a way that was

[1] http://www.beowulf.org
[2] http://setiathome.ssl.berkeley.edu

undetected by their owners. This procedure has generated some controversy. Their method is based on the fact that the transport control protocol (TCP) portion of the TCP/IP network control stack uses ones complement addition to detect errors that sometimes appear in packets as they make their way through data communication networks. This error detection mechanism can be subverted to do logic computations on remote computers that are connected to the net.

Their implementation as well as their basic mechanism was very inefficient and was seen by the authors as an interesting but rather impractical artifact.

Some variations on this method to improve the speed and efficiency have been studied by the author and some work in progress will be discussed. The main thrust of this work is to use the Unix operating system and lower level protocols, those used at the TCP, IP and Ethernet levels, to unwittingly calculate solutions to the 3-SAT problem. The lower the level used, the faster and the more efficient are the solutions obtained. Since many timetabling problems can be mapped into the 3-SAT problem, success with this project could conceivably lead to practical timetabling implementations.

Finally some attention has been paid to using the MAC header of the 802.11 wireless protocol as well as the higher level IP protocol.

References

[Bar95] V. A. Bardadym. Computer-aided school and university timetabling: The new wave. *Lecture Notes in Computer Science*, 1153:22–45, 1995.

[BFJB01] A.-L. Barabási, V.W. Freeh, H. Jeong, and J.B. Brockman. Parasitic computing. *Nature*, 412:894–897, 30 Aug. 2001.

[LWF+00] Q. Liu, L. Wang, A.G. Frutos, A.E. Condon, R.M. Corn, and L.M. Smith. Dna computing on surfaces. *Nature*, 403:175–179, 13 Jan. 2000.

[WXZ04] G.M. White, B.S. Xie, and S. Zonjic. Using tabu search with longer-term memory and relaxation to create examination schedules. *European Journal of Operational Research*, 153:80–91, 2004.