Patrick Griffin
STS.035 Reading #5
March 3, 2004

While computers have proven themselves one of mankind's best tools, interfacing with them has always been difficult. This is true not only in the sense of difficulties programming a machine, telling it what to do in a sensible fashion, but also when dealing with program outputs or even understanding the whole process by which a result is obtained. While machines are useful because they process arithmetic so much faster and more accurately than humans, they are limited to only this capability and thus can formulate abstract things, such as mathematical proofs, only with great difficulty. Thus, computers are good with numbers, and human beings are good with abstraction and pattern-matching, but neither side is particularly good at the other's skill.

Simon describes his efforts, along with Gabe Newell and Cliff Shaw, to make computers process mathematical proofs by the same heuristic mechanism that humans use. This humanistic approach has proven the weakest of the three approaches listed in the first Mackenzie paper, which also mentions non-humanistic brute-force proof methods and human-guided proof methods. The other two mechanisms have been able to prove longer and more difficult theorems than the simple heuristic search because they either better emphasize a computer's number-crunching strength or because they rely on human input for abstract guidance. While the non-heuristic methods may be the most powerful or efficient, they also have verifiability problems, as Mackenzie describes in the second paper. In the case of the four-color problem, a computer proof was disbelieved by much of the mathematical community because it gave no deeper insight as to the meaning of the problem. Essentially, because the computer-assisted proof simply demonstrated (initially) 1936 reducible cases which couldn't be avoided, mathematicians were asked to trust that the program was written correctly. Certainly, hand-checking 1936 cases was infeasible by hand, and doing so would not make any meaning in the solution any clearer. Thus, we see again that computers are good with numbers, humans are good with abstractions or patterns, and neither set is particularly able to fully comprehend the other.

Questions of human vs. non-human algorithm design appear in fields outside of mathematics. One case in point are the placement tools used to translate human-made circuit designs into actual transistor layouts. Human engineers generally create structured designs, where important elements are grouped together into modules – black boxes that perform a particular function. Some placement tools are designed to take these groupings into consideration when placing the transistors, so that related elements are near to each other. Ideally, this grouping would improve performance by reducing communication delays. However, the relationships that group elements in modules do not always imply spacial locality. For instance, a clocking module may contain descriptions of signals that are sent to every corner of the chip, but a placement program that relied only on modularity might place it in the far corner because no one place is better than any other. As a result of their inability to recognize which modules actually imply spacial locality and which don't, most modern placement engines simply flatten the entire design and ignore the man-made grouping information. They then apply a "simulated annealing"

method to optimize placement for performance or area. These methods simply try random placements variations to find which are best. There is no pattern recognition or intelligent guidance involved, the program simply brute-forces through the placement program. Often, this method proves more effective than relying on man-made module locality information, particularly because computers are perfectly willing to keep working on the problem for days. Finally, the most effective placement tools actually mirror the human-guided proof algorithms. In this case, an engineer manually places important modules that the engineer knows require particular spacial locality, and the simulated annealing methods then fill in the rest. While requiring more man-hours and being less flexible than a completely computer based mechanism, human-assisted placement almost always produces better results than either computer-guided module placement or straight simulated annealing.