DISCERNING BIAS IN COMPUTER SYSTEMS

Batya Friedman
Department of Mathematics and Computer Science
Colby College
Waterville, ME 04901, USA
E-mail: b_friedm@colby.edu

Helen Nissenbaum
University Center for Human Values
Robertson Hall
Princeton University
Princeton, NJ 08544, USA
E-mail: helen@phoenix.princeton.edu

KEYWORDS: Computer system design, computer ethics, social implications of computers.

INTRODUCTION

From a study of real cases, we have developed a typology of bias in computer systems. This typology provides a basis for describing, analyzing, and remedying bias in actual systems-in-use. Although other discussions have pointed out bias in particular computer systems, we know of no other comparable work that examines this phenomenon generally and offers a framework for understanding it.

DEFINING BIAS IN COMPUTER SYSTEMS

We can say that a computer system is biased if it both unfairly and systematically discriminates against one group in favor of another. Consider an automated credit advisor that assists in the decision of whether or not to extend credit to a particular applicant. An advisor that systematically denies credit to individuals with consistently poor payment records would not be biased, even though the advisor systematically discriminates against these people. The system is not biased because it is reasonable and appropriate for a credit company to want to avoid extending credit privileges to people who consistently do not pay their bills. By contrast, a credit advisor that systematically assigns poor credit ratings to individuals with ethnic surnames discriminates on grounds that are not relevant to credit assessments and, hence, discriminates unfairly.

The second component of our definition is equally important, for unfair discrimination by itself is not sufficient for bias. In particular, we do not identify as biased computer systems that occasionally and erratically produce unfair outcomes. Consider the example of a traveller who needs to fly from New York to London on the afternoon of November 14. Further imagine that the traveller takes this request to two automated airline reservation systems. The first system, on discovering that three airline carriers can meet this need, say, TWA, American Airlines, and Delta, selects one of the three at random. The second system selects the first of the three airlines from an alphabetical list. Although both reservation systems discriminate in favor of one carrier over the other two, only the second discriminates systematically and, thus, is biased according to our definition.

CATEGORIES OF BIAS

Three overarching categories comprise our typology of bias: pre-existing social bias, technical bias, and emergent social bias. The typology was abstracted and developed from the examination of actual computer systems. We briefly describe the typology and illustrate the categories in terms of two of these systems, the British Nationality Act Program [4], an expert system to determine mechanistically the consequences of the Thatcher government's 1981 British Nationality Act, and the National Resident Match Program [2], a centralized computer system to match medical students to hospital residency programs.

Pre-existing Social Bias

Pre-existing social bias has its roots in social institutions, practices, and attitudes. When computer systems embody biases that exist independently, and usually prior to, the creation of the software, then we say that system exemplifies pre-existing social bias. These biases may originate in society at large, in subcultures, and in formal or informal, private or public organizations and institutions. They can also reflect the personal biases of individuals who have significant input into the design of the system, such as an individual client or system designer. Moreover, pre-existing bias can enter a system either through the explicit and conscious efforts of individuals or institutions, or implicitly and unconsciously, even against our best intentions. For an example of pre-existing social bias, consider the expert system noted above that mechanizes the British Nationality Act. According to many groups [1], the Thatcher government's Nationality Act is at times both racist and sexist. To the extent the expert system accurately represents the Act, whatever bias exists in the Act also will be found in the expert system.

Technical Bias

In contrast to pre-existing bias, the second category in our typology, technical bias, arises from the resolution of technical issues in the design. Sources for technical bias can be found in several aspects of the design process: in the limitations of computer tools such as hardware and software;

in the limitations of algorithms; in the attempt to make human constructs amenable to computers, when we quantify the qualitative, discretize the continuous, or formalize the non-formal; and in the misrepresentation of the system to the user. Consider but one example, also from the British Nationality Act Program, that arose from limitations of the programming language, Prolog, in which the system was implemented. According to the program's authors [4] "Because of limitations imposed for the sake of efficiency...Prolog sometimes goes into infinite loops and fails to prove theorems that are logically implied by the axioms" (p. 377). Due to this limitation, the British Nationality Act Program systematically fails to identify some positive cases of British citizenship. The implication for individuals who fall into this category is obvious: These individuals are entitled to British citizenship but are systematically overlooked by the expert system. Notice that this particular bias does not exist in the 1981 British Nationality Act, but only in the computer implementation.

Emergent Social Bias

While it is almost always possible to identify pre-existing and technical bias in a system design at the time of creation or implementation, emergent social bias emerges only in a context of use: when the system is used in real life, in a real context, with real users. Emergent social bias typically arises as a result of changing societal knowledge, population, or cultural values. An example of this type of bias can be found in the National Resident Match Program. If an individual is training to become a doctor in the United States, chances are that person received a first job placement through this centralized computer system. When the system was originally designed in the 1950s, few married couples -- where both members of the couple were doctors -- participated in the medical match process. Thus, there was little need to develop a fair means for assigning both members of a couple to medical positions. However, beginning with the late 1970s and early 1980s more women entered medical schools and, not surprisingly, more married couples sought medical appointments. At this point, it was discovered that the program's algorithm gave unequal treatment to spouses when both members of a couple sought medical positions [3]. That is, couples were consistently at an unfair disadvantage in the match process. The point here is that the bias against couples only emerged when the social conditions changed.

CONCLUSION

Our typology provides a robust analysis of two actual cases. It also suggests steps a system designer might take in order to minimize bias. Although a thorough treatment requires more space than we have here, nonetheless we offer some brief comments. Minimizing pre-existing bias is a complex business because an astute designer must not only scrutinize the design specifications for bias but must couple this scrutiny with a good understanding of relevant biases out in the world. And even if a designer successfully detects bias in a proposed design, the client may be reluctant to remedy the bias for a variety of reasons. For example, a client might actually want the biases present, as with a "racist" individual who knowingly supports the racial overtones in the British Nationality Act. In turn, minimizing technical bias places an extra demand on a designer to go beyond the technical details internal to a system. A designer must envision the design, the algorithms, the interfaces, in a context of use so that technical decisions do not undermine social values. Indeed, a similar approach is called for in the case of emergent social bias -- perhaps the trickiest of the three types -- for designers must not only envision a system's *intended* situations of use, but realistically anticipate probable extensions of these. At a minimum we recommend that designers clearly articulate constraints on the appropriate situations of a system's use and take responsible action when biases emerge with changes in context. For example, the designers of the National Resident Match Program responded conscientiously to a changing social condition when they modified the system's algorithm to place more fairly dual-career couples seeking residency programs [3]. Finally, we hope our work will spark others' interest in further exploring design methodologies to minimize bias in computer systems.

ACKNOWLEDGMENTS

We thank our research assistant Mark Muir for help with aspects of this analysis. We also extend our thanks to John Mulvey and Deborah Johnson for discussions concerning bias, particularly in relation to the National Resident Match Program. Earlier aspects of this work were presented at the 4S/EASST Conference, Goteborg, Sweden, August, 1992. This research was funded in part by the Clare Boothe Luce Foundation.

REFERENCES

1. Berlins, M., and Hodges, L. Nationality Bill sets out three new citizenship categories (January 15, 1981), *The London Times*, pp. 1, 15.

2. Graettinger, J. S., and Peranson, E. The matching program. *New England Journal of Medicine, 304*, (May, 1981, May), 1163-1165.

3. Roth, A. E. New physicians: A natural experiment in market organization. *Science, 250*, (1990), 1524-1528.

4. Sergot, M. J., Sadri, F., Kowalski, R. A., Kriwaczek, F., Hammond, P., and Cory, H. T. The British Nationality Act as a logic program. *Communications of the ACM, 29*, (1986), 370-386.