Chapter 5. Is Automation the Answer? — The Computational Complexity of Automated Redistricting

5.1. Redistricting and Computers

> "There is only one way to do reapportionment — feed into the computer all the factors except political registration." - Ronald Reagan (Goff 1972)
>
> "The rapid advances in computer technology and education during the last two decades make it relatively simple to draw contiguous districts of equal population [and] at the same time to further whatever secondary goals the State has." - Justice Brennan, in *Karcher* v. *Daggett* (1983)

Ronald Reagan was not the only recent politician or academic to assert that computers can remove the controversy and politics from redistricting. Computers can prevent gerrymandering by finding the "optimal" districting plan, given any set of values that can be specified, claim proponents of automated redistricting. The Supreme Court seems to express a similar sentiment with the emphasis that it put on such mechanical principles as contiguity and compactness in the recent redistricting cases of *Miller* v. *Johnson* (1995) and *Shaw* v. *Reno* (1993).

In Chapter 4, I showed that the mechanical application of compactness standards has previously unanticipated partisan consequences; in this chapter I examine the general automation of the districting process, and its consequences. Will we soon be able to write out a function that captures the social value of a districting arrangement, plug this function into a computer, and wait for the "optimal" redistricting plan to emerge from our laser-printers? I argue that this rosy future is unlikely to be realized soon, if at all, because we are unlikely to solve three problems that face automated redistricting.

In Section 5.2, I show that current redistricting methods are not adequate for the purposes of automated redistricting. Current automation techniques must resort to unproved guesswork in order to handle the size of real redistricting plans. Before automated redistricting produces trustworthy results, large gaps must be filled.

Proponents of automation assume that despite current shortcomings, finding the optimal redistricting plan simply requires the development of faster computers. In Sections 5.3 and 5.4, I show that this assumption is false — in general, redistricting is a far more difficult mathematical problem than has been yet recognized. In fact, the redistricting problem is so computationally difficult that it is unlikely that any mere increase in the speed of computers will enable us to solve it.

Even if these complexities are overcome, automated redistricting faces a serious limitation: To use automated redistricting we must write out a function that meaningfully captures the social worth of districts, and that at the same time can be put into terms rigid enough for computer processing. In Section 5.5, I argue that if we do this we will have to ignore values that are based upon subtle patterns of community and representation, which cannot be captured mechanically.

5.2. Current Research on Automated Redistricting

Although the literature on automated redistricting is at least thirty-five years old, it has seen a recent resurgence. This research generally falls into two categories: The first category addresses the merits of automated redistricting *per se*, and the second category

suggests methods we can use to create districts automatically.[150] In this section I briefly summarize the previous research in each of these two categories.

5.2.1. Arguments for Automating the Redistricting Process

In one of the earliest papers on this subject, Vickrey proposed that districting be automated, and that this automation process be based upon two specific values: population equality and geographical compactness. Under his proposal political actors would be permitted to specify or add criteria to a goal function for redistricting, but they would not be permitted to submit specific redistricting plans. Then plans would be created *automatically,* with no further human input, from census blocks, to meet the goal

---

[150] There is, as well, a third category of literature which indirectly touches on automated redistricting. Authors in this third category typically suggest a *particular* criteria for drawing optimal districts— much of the literature on geographical compactness falls into this category.

In particular, several authors have argued that gerrymandering can be eliminated by drawing districts which are maximally compact (Harris 1964; Kaiser 1966; Polsby and Popper 1991; Stern 1974; Wells 1982). (Also see Young (1988) Niemi et al. (1991) for a survey of other compactness measures.) While these authors focus primarily on the *criteria* for evaluating districts, their core argument is the same as that examined above — i.e., that redistricting can be performed best by automatically optimizing a pre-specified representation function.

function. At its heart, automated redistricting is an attempt to push all decision making to the beginning of the redistricting process.

Vickrey (1961) asserted that automated redistricting provides a simple and straightforward way to eliminate gerrymandering. More recently, Browdy (1990a) followed and extended Vickrey's arguments, and created what seems to be the best case for automated redistricting. Five main arguments are offered in the literature, and these can be easily summarized:

Argument 1. Automated redistricting, in and of itself, creates a *neutral* and unbiased district map (Forrest 1964; Harris 1964; Kaiser 1966).

Argument 2. Automated redistricting *prevents manipulation* by denying political actors the opportunity to choose district plans, while simultaneously producing districts that meet specified social goals (Browdy 1990a; Stern 1974; Torricelli and Porter 1979; Vickrey 1961; Wells 1982).[151]

Argument 3. Automated redistricting *promotes fair outcomes* by forcing political debate to be over the general goals of redistricting, and not over particular plans, where selfish interests are most likely to be manifest (Vickrey 1961).[152]

---

[151] Compactness advocates make a similar argument.

[152] In the compactness literature quoted above, it is argued that the compactness criteria themselves make the automated process fair.

Argument 4. Automated procedures provide a *recognizably fair process* of meeting any representational goals that are chosen by the political process[153] (Browdy 1990a; Vickrey 1961). Browdy also argues that such procedural fairness will help to curtail legal challenges to district plans.

Argument 5. Automated redistricting *eases judicial and public review* because the goals and methods of the districting process are open to view; and because automation process creates a clear separation between the intent and effect of redistricting (Browdy 1990a; Issacharoff 1993) .

5.2.2. Criticisms of Automated Redistricting

Automated redistricting has been criticized as well as praised. Previous authors have raised two central objections to automated redistricting.[154] The first argument against

---

[153] Polsby and Popper (1991) argues similarly that a mechanistic application of formal compactness standards is inherently fair.

[154] There are also a number of papers arguing against particular formal measurements, rather than against automated redistricting. Specifically, compactness standards have been subjected to intense scrutiny. For an introduction to some of the issues surrounding the use of these standards, see Lijphart 1989, Lowenstein and Steinberg 1985, Mayhew 1970 and Chapter 3 in Cain 1984. As most of these arguments are directed against the use of particular geographical criteria and not against automated redistricting in general, I have not included these papers in the preceding summary.

automatic redistricting was originally expressed by Appel (1965). He protested that automated redistricting should not be viewed as inherently objective. He argues that redistricting standards and processes embody political values and that automation of this process hides the fundamental conflict over values. Dixon (1968) as well, pointed out that automated processes, even if based on nonpolitical criteria, may have politically significant results. More recently, Anderson and Dahlstrom (1990) cautioned that political consequences of redistricting goals makes redistricting, whether it is automated or not, inescapably political.

I believe this objection to be both correct and unavoidable. Automation is a process for obtaining a given set of redistricting goals. Neutrality, however, is a function of *three factors*, the process selected, the goals themselves, and the effects of seeking to obtain those goals in a particular set of demographic and political circumstances. There is no general consensus over what "objectively neutral" goals are, or whether they exist[155] at all; therefore, no amount of automation can make the redistricting process "objectively neutral."

---

[155] Much doubt has been expressed as to whether such goals exist. Furthermore, the fundamental conflicts between some of the more commonly proposed goals make such a consensus unlikely. For a discussion of the most commonly proposed goals and some the conflicts between these, see (Cain 1984; Dixon 1968; Grofman 1985; Lijphart 1989).

This objection, however, only applies to the first, and most extreme, claim for automated redistricting. Many proponents of automated redistricting do not make this type of extreme claim, and instead explicitly acknowledge the political nature of redistricting goals. They propose that the automated process be used to neutrally and effectively meet goals generated previously by a political process (Browdy 1990a; Issacharoff 1993; Vickrey 1961). This proposal seems to meet at least the first objection above.[156]

Anderson, echoing Dixon (Dixon 1968), made a second argument against automated redistricting by drawing attention to the legislative process used to select automatically generated plans. He argues that the legislature's willingness to accept the plans that are generated by an automated process will be politically motivated — reintroducing political bias into the process (Anderson and Dahlstrom 1990).

While I believe Anderson to be correct, this specific objection does not seem to me to be strong. If we mandate that the legislature must accept the results of the automatic process, we can prevent this particular attempt to reintroduce bias into the system. In general, however, I believe that researchers have largely underestimated the potential for political biases to become part of the automation process.

---

[156] And, indeed, Dixon, who argues against the neutrality of automated redistricting, freely acknowledges its usefulness for this type of situation (Dixon 1968).

5.2.3. The Core Argument — Automation as a "Veil of Ignorance"

In the arguments for automated redistricting, automation essentially plays the role of a Rawlsian (Rawls 1971) "veil of ignorance" which creates fairness by hiding each actor's position in the final outcome. Like the Rawlsian version, the "veil of automation" attempts to hide the final outcome (i.e., redistricting plans) from those bargaining over the social contract (i.e., redistricting goals and procedures). Like the more general veil of ignorance, the automation process claims to prevent manipulation by promoting a recognizably fair method that will, on average, promote fair outcomes.

Vickrey, in one of the first arguments for automated redistricting, particularly emphasized that in order for the automated process to be successful at promoting fairness, it must be *sufficiently unpredictable*[157] — it should not be possible for political actors to deduce the results of the redistricting goals over which they bargain (Vickrey 1961). This property is essential, for if it does not obtain, then the choice of objective functions collapses into a choice of individual plans, and the incentive to gerrymander

---

[157] Here I use the term "unpredictable" where Vickrey originally used "random." Vickrey's concern was that it should not be obvious to the political actors what exact results derive from a particular value function. Enough randomness in the process would certainly ensure this concern is met, but the process need not be random to do this. For example, if the process is sufficiently *chaotic*, it is not random, but it may still be, for all intents and purposes, unpredictable — satisfying Vickrey's central concern.

remains unameliorated by the automation process. If we can predict the plans that will result from our values, we can pierce the veil of automation.

While proponents of automated redistricting have recognized this need for unpredictability, they have not mentioned the danger from unpredictable results. An automated process for creating districts in accordance with agreed upon values must predictably achieve (or at least approach) the goals that were agreed upon in the bargaining stage, or lose legitimacy.

The automated redistricting process must maintain a delicate balance. To prevent manipulation while maintaining fairness, the automated process must predictably implement the redistricting goals that we have agreed upon in the bargaining process; but it must be unpredictable in every other dimension that is of interest to the bargaining agents. These are difficult requirements to satisfy when the bargaining agents are individuals who narrowly seek specific, hand-tailored gerrymanders. They become even more difficult to meet when bargaining agents represent interest groups or political parties unconcerned with particular incumbents, because such agents are interested in far more general properties of redistricting plans. Can automated redistricting methods reliably produce plans that exclusively embody any specific set of redistricting goals?

5.2.4. Why Current Methods Are Inadequate for Automated Redistricting

Initially, many researchers expressed optimism about the ease of achieving redistricting goals through automation (Nagel 1965; Torricelli and Porter 1979; Vickrey 1961; Weaver and Hess 1963). Vickrey best captures this initial hopefulness:

> "In summary, elimination of gerrymandering would seem to require the establishment of an automatic and impersonal procedure for carrying out redistricting. It appears to be not all difficult to devise rules for doing this which will produce results not markedly inferior to those which would be arrived at by a genuinely disinterested commission." - William Vickrey (Vickrey 1961)

While optimism has now dulled somewhat, because it has been recognized that purely automated redistricting techniques remain generally unsatisfactory (Backstrom 1982), many authors still assume that automation of the redistricting process is within reach (Anderson and Dahlstrom 1990; Browdy 1990b; Polsby and Popper 1991).

In his original paper, Vickrey sketched a method for performing automated redistricting, but did not give develop a precise implementation of this method. Much of the following work assumed the benefits of automated redistricting and focused primarily on providing criteria and methods to use in such automation. Liittschwager (1973) applied Vickrey's method to the Iowa redistricting process. Similarly Weaver and Hess (1963), and Nagel (1965) developed methods and/or measures for drawing districts in accordance with principles of population equality and geographical compactness.[158] More

---

[158] See also Chapter 6 in Gudgin and Taylor 1979 (Gudgin and Taylor 1979), and Papayanopoulos (1973) for a review of early attempts at automated redistricting.

recently, Browdy (1990b) proposed that the method of *simulated annealing* may be generally applicable to the problem of drawing optimal districts.

Many different methods have been used or suggested for finding optimal districts. We can put these techniques into two broad categories: *exact* methods and *heuristic* methods. In the remainder of this section, I review the methods used to search for optimal districts. Although useful for assisting humans, current methods cannot satisfy the goals of automated redistricting.

Limitations of Exact Methods

Exact methods systematically examine all legal districts, either explicitly or implicitly. Explicit enumeration, or "brute force" search methods literally evaluate every district. More sophisticated methods such as implicit-enumeration, branch-and-bound, or branch-and-cut techniques exclude classes of solutions that can be inferred to be sub-optimal without an explicit examination. Finding the optimal districts in this case is then merely a matter of sorting the list of district scores. These methods have been used by several authors to approach very small redistricting problems (Garfinkel and Nemhauser 1970; Gudgin and Taylor 1979, Chapter 6; Papayanopoulos 1973; Shepherd and Jenkins 1970).[159]

---

[159] A close examination of these algorithms reveals that in order to make enumeration complete in a feasible amount of time, "short-cuts" are used where some sub-classes of partitions are *assumed* to be unreasonable, and are disregarded without examination and

Exact methods have two major shortcomings: First, no exact method has been developed that will solve redistricting problems for a reasonably sized plan; and as I will show in Section 4, the mathematical structure of the redistricting problem makes it unlikely that exact methods will be developed in the future that will be able to solve reasonably sized plans. Second, even if we find an exact method that works for real plans, then anyone could use that method to determine the precise plan corresponding to a particular set of redistricting goals; thus, exact methods would have completely predictable results, violating our requirements for an automated redistricting method.

Limitations of Heuristic Methods.

Heuristic procedures use a variety of methods to structure the search for high-valued redistricting plans. None of the heuristic algorithms guarantees convergence to the optimal district plan in a finite amount of time. At best, they are good guesses.

All of the general redistricting heuristics cited in the literature are based upon making iterative improvements[160] to a proposed redistricting plan. The single most

---

without proof of sub-optimality. Restrictions such as “exclusion distance” in Garfinkel and Nemhauser (1970) or limiting examination to “amalgamations” in Shepherd and Jenkins (1970) must be formally classified as heuristic rather than exhaustive.

[160] In the field of computer science, heuristic algorithms are divided into two categories: *iterative improvement,* as above, and *divide-and-conquer* methods. These two

popular method seems to be *hill climbing* and its variants, although a few researchers add more sophisticated features of *neighborhood search* techniques:[161]

• *Hill climbing* methods work by making small improvements on a potential solution until a local optimum is reached. Hill climbing often starts with the current district plan or with a randomly generated plan, and it makes improvements through repeatedly trading[162] census blocks between districts (Moshman and Kokiko 1973; Nagel 1965). Another variant of this method selects arbitrary census tracts to form the nuclei of each district, and then repeatedly adds tracts that most improve[163] the current district until each district is

---

general broad categories include variants such as: simulated annealing and genetic algorithms, which will be described in Section 5.4.

[161] In addition to general redistricting methods, there are a number of special purpose methods of note: Tobler develops an iterative graphical remapping process to generate districts of equal population (Tobler 1973). This process gradually distorts geographical maps to create new maps where population is equivalent to area, facilitating the creation of districts with equal population.

[162] Usually, improvements are made sequentially for each district, but if the number of census tracts is small, several trades may be examined simultaneously.

[163] Often this is simply the tract that is closest to the selected district center (in whatever metric used). The Weaver and Hess algorithm uses linear-programming techniques to select population units to add to the district center.

fully populated (Bodin 1973; Liittschwager 1973; Rose Institute of State and Local Government 1980; Taylor 1973; Vickrey 1961; Weaver and Hess 1963).

• *Neighborhood search* methods are all similar in that they seek to improve potential solutions by examining the value of "nearby" solutions; in this they are similar to hill climbing. Unlike hill climbing algorithms, sophisticated techniques in this class use various techniques to attempt to avoid becoming stuck at local optima. Browdy (1990b) suggests the use of *simulated annealing*,[164] a member of this class of methods.

The main difficulty with all heuristics is that they are, at heart, informed guessing procedures. This is not necessarily bad — when you are faced with a difficult problem, you may be able to find an adequate solution cheaply much of the time by guessing. If we use guessing procedures to decide political questions, however, we must show that our guesses are unbiased and likely to produce good solutions. No researcher in this field has been able to show, either theoretically or empirically, that the districts produced by their methods are near optimal, or that they are unbiased. On the contrary, many heuristic methods produced results that are clearly sub-optimal (Bodin 1973), or depend strongly

---

[164] This and other neighborhood search algorithms will be discussed in more detail in Section 4.

on starting conditions (Browdy 1990b; Nagel 1965; Weaver 1970; Weaver and Hess 1963).[165]

Are the inadequacies of current automated redistricting techniques merely temporary? Will improvements in software design and increasingly powerful computers make automated redistricting easy? In the next section, I will show that automated redistricting has been unsuccessful not only because of current techniques but because of inherent complexities in the structure of the redistricting problem. Furthermore, I will show that these complexities are unlikely be overcome simply through the use of faster hardware or more clever software.

5.3. Automated Redistricting May be Intractable

To find "optimal" redistricting plans, as the advocates of automated redistricting suggest, we must first formulate the redistricting problem mathematical terms and then solve this mathematical problem. In this section, I will show that, regardless of the formulation, the redistricting problem is formally computationally intractable — it is practically impossible to solve exactly.

---

[165] Note that Browdy's proposal for using *simulated annealing* has yet to be implemented. I will discuss this suggestion in detail in Section 5.4.

5.3.1. Redistricting is a Large Mathematical Problem

We can mathematically characterize the redistricting problem in a number of different ways. One simple way to mathematize redistricting is to think of it as a *set partitioning* problem. I will use this particular characterization extensively in this chapter. While there are other characterizations that we could use, such as *graph partition, polygonal dissection* and *integer programming* (See the appendix to this chapter.), the results in this section are not dependent on the characterization we choose, since these characterizations are computationally equivalent. (See Section 5.4.)

In particular, we will characterize redistricting as a combinatorial optimization problem:[166] Imagine that census blocks are indivisible,[167] and that you have complete information about voting and demographic information for every census block in your

---

[166] This in itself is not original to this chapter. Redistricting has been implicitly characterized as a combinatorial optimization problem from Vickrey (1961) onward. See Gudgin and Taylor (1979) and Papayanoupoulos (1973) or previous explicit characterizations of this problem.

[167] Population units are assumed to reflect the most accurate and detailed information practically and/or legally available, unless otherwise specified. For most of this chapter, population units can be read as “census blocks” without too much loss of generality.

state. The redistricting problem is to *partition*[168] the entire set of units into districts such that a *value function* is maximized.[169] This partitioning problem may be complicated by the addition of a set of constraints on districts. These constraints, such as contiguity, may limit the set of legal plans.[170]

---

[168] A partition divides a set into component groups which are exhaustive and exclusive. More formally:

For any set $\mathbf{x} = \{x_1, x_2, \ldots, x_n\}$, a *partition* is defined as
a set of sets $\mathbf{Y} = \{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_k\}$ $s.t.$
(1) $\forall x_i \in \mathbf{x},\ \exists \mathbf{y}_j \in \mathbf{Y}, s.t.\ x_i \in \mathbf{y}_j$
(2) $\forall i, \forall j \neq i, \mathbf{y}_j \cap y_i = \varnothing$

[169] More formally:

Given:
• a set of census blocks $\mathbf{x}$
• the set of all partitions of $\mathbf{x}$, Y
• a value function on partitions, V(y)
The optimal district plan is
$D^* = \max_{y \in Y}(V(y))$

[170] These can be represented as formal constraints on membership in the set of allowable partitions in note 17 above, or may for some approximations, simply be incorporated in the value function to be optimized.

The redistricting problem poses special difficulties because the size of the solution set can be enormous. In general, it will be impossible to attack the problem by a brute force search through all possible districting arrangements.

Formally, the total number of distinct[171] plans that can be created using *n* population blocks to draw *r* districts is characterized by the function:[172]

$$S(n,r) = \frac{1}{r!}\sum_{i=0}^{r}(-1)^{i}\left(\frac{r!}{(r-i)!\,i!}\right)(r-i)^{n}$$

Even under the assumption that each district is composed of exactly *k* population blocks[173] (hence, $r = n/k$) the number of possible plans is still a rapidly growing function:

$$S'(n,r,k) = \frac{n!}{k!(r!)^{k}}.$$

The magnitude of this problem is often not fully recognized. For even a small number of census tracts and districts, the number of possible districting arrangements

---

[171] This number reflects districts that are distinct, ignoring the numbering order of districts. Merely renumbering the districts without changing the composition of at least one district does not result in a different plan.

[172] *S* is known as a "Stirling Number of the Second Kind." See Even (1973) for a good introduction.

[173] Which is not correct, but closer to the real situation than the formula above.

becomes enormous. As an example of this, Table 5-1 lists the number of possible plans that could be used to divide a small hypothetical state into two districts:

Type of Population Block	*Total Number of Blocks*	*Blocks per District*	*Number of Plans*
counties	10	5	945
census tracts	50	25	$5.8 * 10^{31}$
census blocks	250	125	$4 * 10^{245}$

Table 5-1. Number of plans available to divide a hypothetical area into two districts, by type of population block.

As Table 5-1 shows, the size of the redistricting problem grows rapidly as a function of the number of population units being used. In fact, this table understates the size of the problem, because it assumes all districts have an identical numbers of blocks.

The number of districts, *r*, is also an important factor in determining how many plans are possible. The number of plans possible will increase in *r* up to a point, and then decrease, as Table 5-2 shows:

Number of Districts	*Blocks per District*	*Number of Plans*
1	24	1
2	12	$3.2 * 10^{11}$
3	8	$9.2 * 10^{12}$
4	6	$4.5 * 10^{12}$
6	4	$9.6 * 10^{10}$
8	3	$1.6 * 10^{9}$
12	2	$1.3 * 10^{6}$
24	1	1

Table 5-2. Number of Plans possible when dividing 24 population blocks evenly into *N* districts.

An exhaustive search to find the optimal plan will be impractical for all but the coarsest population units and extreme number of districts per census block. Only by using a large population unit, such as a county, for our "indivisible" units, can we make exhaustive search manageable. Unfortunately, using such coarse granularity is likely to substantially decrease the quality of our solutions. Furthermore, use of such coarse population divisions is unlikely to lead to solutions where even rough population equality is maintained between districts. If we want to draw districts using accurate, fine-grained population units, such as census tracts or blocks, the number of plans involved makes exhaustive searches unmanageable.[174]

Several proponents of automated redistricting, when faced with a prohibitive number of possible plans, suggest that other, nonexhaustive procedures be used to generate districts (Nagel 1972; Papayanopoulos 1973). Certainly, exhaustive search is not necessarily the only method guaranteed to find optimal districts. However, in the remainder this section, we will show that *any method for finding optimal districts* is likely to be computationally hard, and thus impractical for all but the "smallest" redistricting problems.

---

[174] For a state such as California, where 100,000 census blocks must be assigned to 50 districts, if started at the creation of the universe, a computer that could examine a million districts a second would still not be finished.

5.3.2. Candidate Value Functions

The difficulty of solving the redistricting problem will depend upon the particular value function and constraints we use. In the next section, I summarize the most common candidates for value functions before analyzing the difficulty of the redistricting problem for each one.

While there are practically no political values that are not subject to debate, a number of criteria are commonly thought to be good candidates for redistricting goals. Grofman and Lijphart summarize these, and I list the five most common types below (Grofman 1985; Lijphart 1989):

1. *Population equality* between districts is believed by many scholars to be necessary for political fairness.

2. *Contiguity* has received much attention in combination with compactness.

3. *Compactness*, which attempts to capture the geographic regularity of districts also appears in many state constitutions. Compactness has been defined in many different ways. (See Niemi et al. (1991), for a survey of these.)

4. *Creating fair electoral contests* is another criteria that is sometimes found in state constitutions. Of course, there are many possible definitions of a "fair contest": including *maximal competitiveness* (maximizing the number of close elections),[175] *neutrality* (which

---

[175] Here I group together a number of different types of criteria including: "electoral

specifies that the electoral system should not be biased in favor of any political party in awarding seats for a certain percentage of the vote), and the goal of a *constant swing ratio* (seats/votes share) for each party.

5. The last set of common redistricting goals dealing directly might be termed *representational* goals, as they are difficult to formulate without referring to a concept of representation. These include *protection of communities of interest* and *nondilution* of minority representation.

As well as being theoretically and philosophically important, these values often carry the weight of law (Grofman 1985): The U.S. Supreme Court has found the constitution to require *de minimis* population deviations between Congressional districts and only somewhat larger deviations between state legislative or local government districts. Furthermore, 37 states require districts to be contiguous, 24 states require compactness[176] and 2 states require what might be loosely interpreted as an electoral response function.[177] Representational goals also have some legal force: Protection of communities

---

responsiveness," "neutrality," "competitiveness," and "constant swing ratio," which are often addressed separately in the literature.

[176] Only three states formally define "compactness."

[177] These are vaguely defined in the constitutions of these states as directives to "not unduly favor any person or political party (faction)."

of interest is required in five states, and nondilution of minority interests is required under the Voting Rights Act.

Despite the relative popularity of the four types of goals above, there is no political or academic consensus over them. Nor can formalization or automation somehow make the goals "objective" — the political consequences of redistricting goals will still exist. Although many have recognized this before, it cannot be emphasized too strongly that at best, automation can neutrally implement these goals, once they have been decided upon by a *political* process.[178]

5.3.3. What is a Computationally Hard Problem?

In this section I will show that for any of the aforementioned value functions (or combinations of them), the problem of finding an optimal districting plan is computationally complex - any attempt will probably be thwarted by the size and complexity of the redistricting problem. To prove this result I will have to introduce some formal definitions from computational complexity theory.[179]

---

[178] In this point I agree with two of the main proponents of automated redistricting (Browdy 1990a; Issacharoff 1993).

[179] In order to present the next set of results, it is necessary to define a number of terms and ideas referring to problems, solution methods, and solution complexity. The length of this chapter necessitates that this section be limited to what is essential for

Computational complexity (or "structural complexity") theory and the related field of computability theory are two branches of theoretical computer science. These disciplines are devoted to analyzing the difficulty of solving specified discrete problems using computers.[180]

Researchers in computer science and in operations research use computational complexity theory extensively when they analyze problems. While this type of analysis has been adopted only recently by political scientists, computational tractability is becoming recognized as a prerequisite for practical electoral rules[181]: Kelly (1988a, 1988b) analyzes the complexity of a number of voting rules, and he establishes some conditions for computable electoral rules (Kelly 1988a; Kelly 1988b). Bartholdi, Tovey and Trick analyze the complexity of manipulating elections; they argue that while almost

---

understanding the result. For a more lengthy and formal characterization, see Papadimitriou (1994).

[180] For a review of recent developments in this field, see Book (1994).

For problems that are (unlike partitioning) continuous, rather than discrete, the field of "information-based complexity" also has relevance. For an introduction to this latter field, see Traub and Wozniakowski (1992).

[181] Also see Deng and Papadimitriou (1994) on the complexity of different cooperative solution concepts that are used in some positive political theory models.

all electoral rules are theoretically open to manipulation, some rules may be practically impervious to manipulation because of the complexity of the calculations a manipulator would have to perform (Bartholdi, Tovey and Trick 1989; Bartholdi, Tovey and Trick 1992).

Basic to computational complexity theory is the definition of a problem. A *problem* is a general question to be answered. In the case of redistricting, the problem is to find the districting plan that maximizes our value function — formally, we must find the optimal partition.[182] I will use the term *redistricting sub-problem* to distinguish the case where we have pre-specified a particular value function, such as compactness, rather than taking the value function itself to be a parameter. Hence, redistricting to maximize (a particularly formally defined measure of) population equality is a sub-problem. A problem possesses several *parameters*, or free variables. For any redistricting sub-problem, the parameters consist of the population units from which we are to draw the plan and the vector of values assigned to those population units. An *instance* of a problem is created by assigning values to all parameters. Finding the arrangement of

---

[182] We can characterize redistricting either as a general problem where the value function itself is a parameter, or as a class of similar partitioning problems, each with a separate value function. Our choice of characterization does not affect the results found below.

Iowa's 1980 census tracts that maximize population equality would then be an instance of a redistricting sub-problem.

The second set of terms refers to solutions to the preceding problems. An *algorithm* is a general set of instructions, in a formal computer language that, when executed, solves a specified problem. An algorithm is said to *solve* a problem if and only if it can be applied to *any* instance of that problem and is guaranteed to produce an exact solution to that instance. To continue the example above, an algorithm would be said to solve the population-equality redistricting problem only if it were guaranteed to find a population-equality-maximizing solution for any set of census blocks that we put into it.

The final set of terms refers to properties of problems and their solutions. A problem is said to be *computable* if and only if there exists[183] an algorithm which solves the problem.[184] For computable problems we define the *time-complexity* (hereafter

---

[183] Here I use the term *exists* in the formal, mathematical sense — we do not necessarily have to know which algorithm solves a problem to show that such an algorithm must exist.

[184] Turing (1937) (1937) showed that there are problems for which solutions exist that are not computable in the sense used above. I am not arguing that practical redistricting criteria are likely to be noncomputable, although this is a theoretical possibility.

abbreviated to "complexity") of an algorithm to be a function that represents the number of the instructions that algorithm must execute to reach a solution.[185] The complexity of an algorithm is expressed in terms of the *size* of the problem, roughly equivalent to the number of input parameters.[186] The size of redistricting is simply the number of population units that are used as input. An algorithm is said to take polynomial time if its time-complexity function is a polynomial and is said to take exponential time, otherwise.[187]

---

[185]This definition assumes a serial (single processor) computation model, but the results are not altered if we use parallel-processing: The sum of the time needed by a set of parallel-processors to solve a problem can be no less than the total required in the serial model.

[186] The time complexity of an algorithm is conventionally denoted as $O(f(n))$ where n is the size of the problem. Additive and multiplicative constants are omitted, as these vary with the computing model used. Thus the number of steps to solve an algorithm of $O(n)$ complexity is a linear function of the number of inputs.

[187] In addition to analyzing the time required to solve a problem, we can formulate analogous tractability criteria for the storage space requirements of a problem (or for practically any other of its resource requirements). It can be shown that problems that require exponential space will also require exponential time, but not vice-versa. Fortunately, none of the redistricting sub-problems discussed here needs exponential

A problem is often said to be *computationally tractable* if there exists an algorithm which is of polynomial complexity for all instances and which solves the problem. Conversely, a problem will be said to be *computationally intractable* (also "computationally complex," or "computationally hard") if the (provably) optimal algorithm for solving the problem cannot solve all instances in polynomial time.

Although we have defined complexity in terms of time, we may usefully think of it as a measure of *cost* as well. If time is costly, and if there are no exponential economies of scale associated with time, computationally intractable problems will be prohibitively expensive, since the cost to solve such problems will also grow at an exponential rate.[188] Obviously, the time-costs of a redistricting are unlikely to exhibit exponential economies of scale. If anything, wasted time will likely exhibit constant or negative economies of scale: if redistricting takes too long, it will start to disrupt elections seriously.

This characterization of problem difficulty has two main strengths. It is independent of any particular computer hardware design technology and it classifies the difficulty of the problems themselves, not of particular methods used to solve these problems.

---

space.

[188] Alternatively, if we were to use parallel-processors to solve the problem, the *number* of computers would grow exponentially (at least)— thus our costs still grow exponentially.

First, results under this characterization are *implementation independent.* Different computer languages (and encoding schemes for the parameters) may alter the time complexity of an algorithm, but no "reasonable"[189] language will be convert a

---

[189] All known, physically constructable, computer architectures are *"reasonable"* in this sense, and it is believed that all possible computers based on classical physical principles preserve this property (Papadimitriou 1994). However, there is some debate over whether this implementation independence applies to hypothetical computers designed to utilize unexplored properties of quantum physics. Two authors, in particular, assert that the above model does not accurately describe all potential problem solving devices.

Deutsch asserts that under the "many-universes" interpretation of quantum theory, one could design a device to exploit an infinite number of alternative universes for parallel calculation (Deutsch 1985). Under this controversial interpretation of quantum theory, devices may be built which would be able to compute some (but not all) problems in polynomial time that are computable on all conventional computers only in exponential time (Deutsch and Jozsa 1992).

Penrose (1989; 1994) makes a somewhat different argument, asserting that currently unresolved areas of quantum physics may provide fundamentally different ways of solving problem than is represented by the Turing model. Penrose argument, which is too rich and

polynomial algorithm to an exponential algorithm. While a more powerful computer may be able to perform each atomic operation more quickly, it will not alter the time complexity function of the problem. Intractable problems cannot be made tractable through improvements in hardware technology.

Second, results under this characterization apply to the problem itself, not to a particular method used to solve this problem. Problems which are shown to be difficult under this characterization are difficult for *any* possible computer method. Since it is the problem, itself that requires exponential time, these problems cannot be made tractable through advances in software or algorithmic design.

This characterization is also subject to several important limitations. These limitations have caused its use as an absolute measure of problem difficulty, especially for social science problems, to be justly criticized (Page 1994). I will briefly summarize these limitations here, and in Section 5.5 I will extend the analysis to address the relevance of these limitations for the redistricting problem.

First, the distinction between tractable and intractable problems is most important for instances of large size - where the exponential factors in the time requirements of these problems become dominant. Consider the following two problems. Problem “A” is

---

detailed to be adequately summarized here, asserts not only that quantum physics allows mechanisms for problem solving which are fundamentally different from those used in today’s computers, but that the human brain actually employs such mechanisms.

computationally intractable and takes $O(1.1^n)$ steps to solve. Problem "B" is computationally tractable and takes $O(n^{14})$ steps. Although the time needed to solve problem A will eventually become much greater than the time required for problem B, for problem sizes less than one thousand, we can actually solve problem A much more quickly.

Second, when we use this characterization we require that problems be solved exactly. Some problems that are computationally difficult to solve may be *approximated* much more quickly. If the approximation reached is (provably or empirically) close enough to the optimal solution to the problem, for practical purposes we may not need to find the exact best solution.

Third, when we use this characterization we require that our algorithms *always* reach a *correct* solution for every problem instance, requirements that make computational complexity a function of the worst-case problem instance. Since we base our analysis on the worst-case, we may overstate the complexity of the problem on average. Furthermore, since we require that our solution-algorithm neither make errors, nor give up on a problem, we will drop from our analysis some algorithms that are probabilistic. While such algorithms do not formally "solve" a computationally hard problem, they may be quite useful if their rates of error and of failure are sufficiently low.

The three caveats above offer to us possible escape routes around computationally intractable problems, but these are only *possible* routes. And as I will show in Section

5.5, in general the requirements of automated redistricting procedures make these avenues unlikely to be fruitful.

5.3.4. Redistricting is a Computationally Hard Problem

In the previous sections, I showed how redistricting is deeply connected to mathematical partitioning problems. Many researchers in computer science have examined partition problems and reached some conclusions about their computational complexity. In this section, I show that the redistricting problem in general, and even many simpler redistricting sub-problems are likely to be intractable.

Proving that a problem is intractable is difficult — researchers have been unable to determine whether most problems are tractable (Papadimitriou 1994). There are, however, a number of large classes of problems that computer scientists believe to be intractable. The oldest of these is called the class of *NP-complete* problems.

Cook defined the first NP-complete problem (Cook 1971), which has now been shown to belong to a large set, consisting of hundreds of problems in many fields. Karp characterized the most important property of NP-complete problems (Karp 1972): *polynomial-time reducibility*. Any NP-complete problem can be transformed into any other NP-complete problem in polynomial time.[190] Thus, if you could prove that any NP-

---

[190] Polynomial reductions are defined so as to preserve space complexity characteristics as well. Furthermore, there is an even deeper equivalence between all

complete problem is formally intractable, you would have proved all such problems intractable, and vice-versa.

Search for a proof of the intractability of NP-complete problems has been of the most famous open problems in computer science for over two decades. While no proof of intractability has been found, no polynomial algorithms have ever been found that solve any of these problems, and because of the breadth of the class of problems, it is widely believed that no such algorithms exist.

The class of NP-complete problems is not the only class that is believed to be intractable — there are many other classes of problems that are equivalent to each other but not to problems in the NP-complete class. For our purposes, however, we need consider only the NP-complete class and the related class of *NP-hard* problems: The *NP-hard* class is a superset containing the *NP-complete* class; this class is potentially harder to solve than NP-complete problems because although if any NP-complete problem is intractable, then all NP-hard problems are intractable, the reverse is not true.[191] The diagram below illustrates the probable relationship between the NP-complete, NP-hard and tractable classes of problems. (Figure 5-1)

---

known NP-complete problems — each problem can be transformed to any other by a simple functional mapping (technically a “bijection”).

[191] Any NP-Hard problem can be shown to be NP-complete for at least some instances, but not necessarily for all instances.

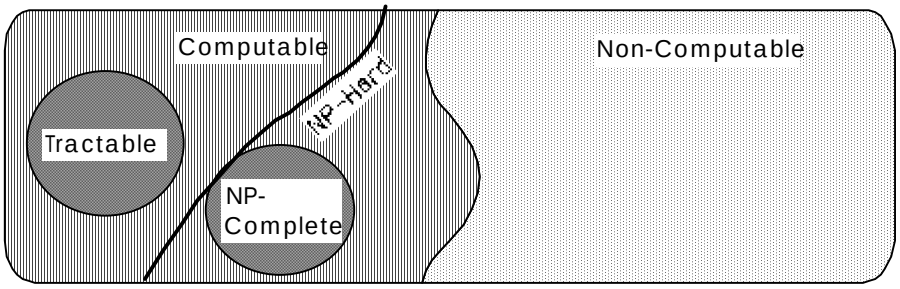


Figure 5-1. A diagram showing NP-Complete and Related Complexity Classes

In the appendix to this chapter, I show formally how the most common redistricting sub-problems, such as finding the optimal set of compact districts, are NP-complete or NP-hard.[192] Table 5-3 summarizes these formal results; for each sub-problem I give a short example of its formal characterization, and a reference to the corresponding problem in complexity theory.

---

[192] This finding contradicts Garfinkels's and Nemhauser's (1970) (Garfinkel and Nemhauser 1970) prediction that the time required for their integer programming technique should be expected to be approximately linear in population units.

Redistricting Sub-Problem	*Example Characterization*	*Reference Problems*
Equal population districts	*set partition*: divide members of set into subsets minimizing value function • population unit: represented by an integer-valued set member • district: each subset in partition is a district • value function: the largest difference in population between any two districts	*Weighted Set Partition* (Karp 1972) *3-Partition* (Garey and Johnson 1983) *Weak Partition* (Johnson 1982) *Integer Programming* (Garey and Johnson 1983)
Compact districts	*set partition:* as above • population unit: represented by a vector valued set member • district: as above • value function: (a) maximum distance between points in a district (b) total perimeter of districts	*Distance-d Partition of Points in the Plane* (Johnson 1982) *Minimum Perimeter Partition into Rectangles* (Johnson 1982)
Competitive districts	*set partition:* as above • population unit: represented by a weighted set member. Weight is Republican registration - Democratic registration in each districts. • district: as above, district weight = sum of weights of the population units it contains • value function: minimize sum of squared district weight	*Minimum Sum of Squares* (Garey and Johnson 1983)
Contiguous districts (with population equality)	*graph partition*: divide nodes of a graph into connected subsets minimizing value function • population unit: represented by a node of the graph • contiguity relationship: population units contiguous to each other are connected by edges in graph • district: as above • value function: the largest difference in population between any two districts	*Cut into Connected Components of Bounded Size* (Johnson 1982; Johnson 1984)

Table 5-3. Redistricting Problems that are at least NP-Complete

While I refer to a number of particular characterizations in Table 5-3, it is important to realize that none of my results in this chapter is dependent on the use of a particular

characterization.[193] Since all of these characterizations are in the set of NP-complete problems, they are all formally equivalent — any algorithm that solves one problem can be simply reformulated to solve any other without a change in complexity class. The advantage of multiple characterizations is that each characterization may suggest various limitations that can be put on the problem that may make the problem *easier* to solve or approximate. (See Section 5.4 below.)

While I have focused on sub-problems, you should note, as well, that these complexity results apply to the general redistricting problem. Since complexity results describe worst case properties, my results demonstrate that the redistricting problem, *in its most general form,* is at least as difficult as any NP-complete problem.[194] A further

---

[193] Reformulating the redistricting problem in other ways, can be useful if it suggest natural restriction that can be put on the problem in order to simplify it. Restrictions that re natural in one context, such as planarity or graphs in the graph-partitioning problem, may not be obvious when the problem is formulated as an integer partition problem.

[194] Each of the redistricting sub-problems above is an instance of the redistricting problem. If some instances of the sub-problems require exponential time, then the general problem as well must also require this amount of time. Hence, the redistricting problem has a time complexity at least as large as its sub-problems. We have not shown that these sub-problems represent worst cases of the redistricting problem, however, so it is possible that some instances of the general redistricting problem may be worse than NP-complete,

implication of these results is that finding an optimal district under any *combination* of these "hard" goals is also hard. In sum, we should not expect automated optimal redistricting to be tractable for an arbitrary choice of population units, number of districts, and an arbitrary value function.[195]

---

or even unsolvable.

[195] When making the claim that automated redistricting is formally intractable, it is sometime objected that this claim cannot be true because political actors are able to gerrymander so well. If political actors can gerrymander optimally, the argument guess, then so should computers. I believe this objection is based on three false assumptions:.

• Claim one: *Humans already perform (near) optimal gerrymandering*. There is little evidence for this claim; in fact, there is much evidence to the contrary, and many examples of attempted gerrymanders that have had far from the intended results.

• Claim two: *Automated redistricting is no more difficult than gerrymandering.* Again, the available evidence seems to point toward the opposite conclusion. Automated redistricting is significantly more complicated than gerrymandering in three important respects. Gerrymandering usually involves the maximization of one simple goal. Optimal redistricting may involve many simultaneous, complicated, and conflicting goals. Gerrymandering is often limited to relatively small modifications of an existing plan. Automated redistricting processes must examine a much wider range of possible plans. Finally, gerrymanders need not be optimal to be politically effective. The social value

It is also important to note that the redistricting goals I have listed in Table 3 are common, straightforward, and relatively simple. For example, of all the redistricting goals specified in the literature, population equality is probably the most straightforward to quantify, and the easiest to evaluate.[196] Yet even the task of drawing a district plan to minimize population deviation alone is computationally hard.

---

function being optimized by an automated process may be much more sensitive to suboptimality.

• Claim three: *If human's can perform a (mathematical) task well, computers can (at least theoretically) perform the same task as well.* The lack of success in the field of artificial intelligence is evidence to the contrary. See Dreyfus 1992 (Dreyfus 1992) for a review of the successes and failures in this field, and a more detailed argument against claim three. Even advocates of artificial intelligence recognize that such tasks as recognizing human social and political relationships, and applying "common sense," are among the hardest problems for computers. (See Chapter 3 of Taylor 1989.)

[196] Even this goal can be defined in a number of ways, depending on how you wish to measure the inequality between two districts. For instance, we might focus on the maximum differences between the largest and smallest districts, or their ratios, or their average differences, etc.

Still, this does not demonstrate that the redistricting problem is hopeless. Section 5.5 explores the avenues available for managing intractable problems and the promise of these avenues for redistricting.

5.4. Attempts to Escape the Intractability of Automated Redistricting

In Section 5.4 of this chapter I demonstrate that the redistricting problem is NP-complete, and we briefly reviewed several caveats to our definition of intractability. Do these caveats allow automated redistricting to escape intractability?

Recall the definition of "tractability": A problem is tractable only if a polynomial-time algorithm exists which is *guaranteed* to *exactly* solve *all* instances of this problem. Are we being overly demanding? If we demand somewhat less of our solution algorithms, can we find practical solutions to automated redistricting problems?

NP-completeness is a limited form of intractability,[197] and there are a number of possible avenues for dealing with these types of problems. In Section 5.4 we did not

---

[197] NP-complete problems certainly do not encompass all intractable problems, nor are they the "worst" class of intractable problems. The intractability of NP-complete problems takes a limited form.

A problem may be intractable for a number of different reasons. I list three here, in order to illustrate several levels of problem difficulty. First it may be that the solution itself

demonstrate that *all* possible redistricting sub-problems are intractable: If we restrict our redistricting values sufficiently, we can certainly find a tractable way of finding optimal

---

is unmanageably large or otherwise unmanageable. Second, it may be that the solution in itself is manageable, but is difficult to both to solve and verify. Third, in the least difficult case, only finding the solution is difficult — once found it can be easily verified and put to use.

An example of a problem of the first type, where the solution itself is unmanageable is: "enumerate the set of all possible district plans." Simply printing this solution for this type of problem is untenable for all but the most minuscule instances. This type of problem is provably intractable. Fortunately, problems of this type are in some ways uninteresting, in that, even should a solution for the problem eventually be obtained, one would be unlikely to be able to make use of it.

Problems of the second type are esoteric and too awkward to describe here — see Bellare and Goldwasser (1994) for a description.

NP-complete problems fall into the third category. An example of a problem in the third type is: "Does there exist a district plan where the maximum population difference between districts is less that *B*, and if so, specify that plan?" For this problem, as long as computation of the value function *f* is tractable, finding a satisfactory plan may be difficult — but given a satisfactory plan, it is easy to verify that the value of the plan exceeds *B*.

districts under them. Similarly, if we restrict the set of inputs to our problem enough, we may find it quite tractable.

Furthermore, we did not in eliminate all *practical* approaches to general redistricting problems. If the problem cannot be made tractable through restricting our values or data, we might be able to find a probabilistic method that solves the problem most of the time, or a deterministic algorithm that works well on most cases. Alternatively, we may be able to find a method that quickly finds an approximately-optimal solution.

We must, however, be cautious; although these escape routes are open in theory, they pose a number of technical and political difficulties. In the remainder of this section, I will explore these possible escape routes.

5.4.1. Solving NP-Complete Problems — An Example

While all NP-complete problems are reducible to each other, in a practical sense they are not all equally hard to solve: Some problems can be easily restricted, answered probabilistically, or closely approximated.

We can use a simple example to show the practical difference between two formally intractable problems. For this example, imagine that a rich acquaintance has died and that you have been asked to be executor of the estate. The rich acquaintance was a collector of art and antiques, and the estate is made up entirely of unique and valuable items. You are told that you must divide the estate between its inheritors in such a way that (1) you give away all the items (you cannot sell them and give away the money), (2) each item goes to a single inheritor, and (3) you must maximize the *subjectively equality* of the

division — the monetary value each person assigns to her own share must be as close as possible to the value another person assigns to his own share.

Formally this is an NP-complete problem,[198] but if all inheritors share common values for each object (perhaps they are all antique dealers who know current market prices), this problem is not very "hard" for many practical cases: If there are only two inheritors, and no item is "priceless," we can find the optimal solution in polynomial time using dynamic programming (Garey and Johnson 1983). On the other hand, if there are more inheritors, but all the objects are of approximately equal value, we can easily minimize the inequality between shares.[199]

Now suppose that each inheritor has differing private values[200] for sets of objects; it is especially difficult when the relevant sets differ for each inheritor as well. (For

---

[198] It is an example of the weighted set partition problem (Garey and Johnson 1983).

[199] Even if the antiques differ greatly in value, we may be able to get relatively close to an even division. While I can find no approximability results for this partition problem, the following algorithm is quite successful for bin packing, a similar problem: Simply order all the objects from least to most valuable, then assign each object in this list to each inheritor in turn, until the list is exhausted (Garey and Johnson 1983).

[200] Here I disregard the additional difficult of eliciting the inheritor's private values for each set of items — the problem is difficult even if you are completely informed about

example, aunt Martha attaches great sentimental value to the sofa and end-table combination, whereas uncle Henry hates the end-table but has always coveted the sofa and matching wall-hanging.) If you have a lot of antiques to distribute, although the problem remains much the same in structure to our first problem, it has become much more difficult to deal with practically.

5.4.2. Problem Size and Computational Complexity

Even problems which are computationally hard may be solved easily for sufficiently small cases. Are the sizes of the redistricting problem which are typically encountered small enough that intractability is not a practical issue?

Unfortunately, the answer to this question is probably no. As Section 3 shows, the number of possible solutions to a problem grows both as a function of the number of districts being drawn (to a limited extent) and as a function of the number of population blocks that we use. As the appendix shows, the time necessary to solve redistricting sub-problems grows exponentially as a factor of both districts and population blocks.

While the number of districts is typically reasonably small, ranging from one to approximately fifty districts in the United States, the number of population blocks in practical cases is quite large — ranging from several thousand to approximately one hundred thousand. Although we can only approximate the time needed to solve these redistricting problems, the number of solutions that must be searched is large enough that

---

each inheritor's basic private values.

we can reasonably expect the exponential time-requirements of the algorithm to be dominant, even if the exponential growth is relatively small.[201]

5.4.3. Restricting the Redistricting Problem

It is well understood that any problem, when sufficiently generalized, becomes computationally hard, and conversely, any sufficiently restricted problem becomes trivial. Because of this common-sense principle, we must consider whether there are any natural restrictions that we can place on the redistricting problem that would make it tractable.[202]

There are three basic types of restrictions that we can place on the redistricting problem: First, we can restrict the redistricting goals that we will consider. Second, we can rule out some redistricting plans, such as those containing noncontiguous districts, as

---

[201] An exponential growth factor as low as 1.001^n is likely to make such large problems intractable.

[202] Note that here I use *restriction* to refer to how we limit the goals and inputs that we will consider for redistricting problem itself. These restrictions do not necessarily correspond to *constraints* on districting that limit electoral manipulation (i.e., "gerrymandering"). For example, restricting the automated redistricting problems to consider only county-level population data, rather than finer-grained population data, might actually make electoral manipulation easier.

illegal — we can throw them out of the analysis. Third, we can restrict the data, or population units, that we feed into our automated redistricting process.

For purposes of clarity, we will discuss the various types of restrictions separately. In practice, restrictions of all types are often combined to simplify problems and avoid computational complexity. But although we can reduce the practical difficulties of finding a plan by combining multiple restrictions, we cannot use these combinations to eliminate the political and normative problems we have already encountered.

Restrictions on Value Functions

We may make the computer's task easier by limiting the number and types of goal function that it has to optimize. By using such restrictions, we may be able to take advantage of specially tailored optimization algorithms, or we may be able to eliminate exponential time requirements for more general optimization algorithms.

While there must exist goal functions that are "easy" to optimize,[203] there are two difficulties with this approach. The first difficulty is a purely practical one — reasonable candidates for redistricting goals seem to be computationally difficult to optimize. As we saw in Section 4, even the simplest and most popular value functions are *all*

---

[203] For example, a function that assigns the same constant value, although completely uninteresting, is easy to optimize. Unfortunately, there seem to be no interesting value functions that are "easy" to optimize on unrestricted inputs.

computationally hard to optimize. Furthermore, any (simply weighted) combination of these functions with each other, or with any other function, will also be computationally hard. In sum, our present goals do not lend themselves to easy computation.

The second difficulty is normative and political. Proponents of automated redistricting argue that one of the strengths of automation is that it allows goals to be decided by a political process — automation should only affect the implementation of goals, not their choice. This separation of goal choice and implementation is violated by placing restrictions on allowable redistricting goals. Allowing the limitations of a computerized process to restrict, *ex ante,* the type of representational goals that society is allowed to pursue may be normatively and politically unacceptable.

Restrictions on Input

Restrictions on input can also, theoretically, allow us to escape intractability. In this chapter, as in most automated redistricting research, the basic inputs have been defined in terms of vector-valued *population blocks* such as census blocks. We can place restriction on census blocks directly or indirectly. Restrictions on these population blocks can be used directly and indirectly for this purpose.

We can use limitations on input *directly* to reduce the computational complexity of a problem by eliminating its worst cases — if we can predict what cases create problems for our redistricting algorithms. For example, if we want to create district plans solely to minimize population inequalities, we can simplify this problem by restricting all

population blocks to be equal in size.[204] In this restricted case, any of the heuristics listed in Section 5.2 will quickly find an optimal solution to this problem. While not completely trivializing the problem, this eliminates the possibility of cases which might cause the algorithm to take exponential time to complete.

For more complicated problems, we can use input restricting indirectly, and in combination with restrictions on redistricting goals, to reduce the number of solutions that an optimization algorithm needs to consider. Methods such as "branch and bound" and "cutting planes methods" use different applications of this general principle (Balas and Toth 1985; Reinelt 1991). Balas gives a succinct description of these methods:

> "Enumerative (branch and bound, implicit enumeration) methods solve a discrete optimization problem by breaking up its feasible set into successively smaller subsets, calculating bounds on the objective

---

[204] A similar alternate strategy for drawing equal population districts is to arbitrarily split population-blocks in order to make them approximately equal. This method seems to be one of the most popular in the political arena, where it is not uncommon to find districts that are based on census blocks or tracts, but which splits these in order to maintain equality between districts. Since in this chapter "population unit" has been assumed to reflect the finest grained unit for which census information is available - splitting population-blocks must be regarded as an approximation method. The limits of approximations methods are discussed in Section 5.2 below.

> function value over each subset, and using them to discard certain subsets from further consideration. The bounds are obtained by replacing the problem over a given subset, with an easier (relaxed) problem, such that the solution value of the latter bounds that of the former. The procedure ends when each subset has either been reduced to a feasible solution, or has been shown to contain no better solution than the one already in hand. The best solution found during he procedure is a global optimum."

These methods are usually very sensitive to our choice of goal functions — we must know quite a lot about the goal function to derive the "relaxed" problem that we use to set bounds.[205] We must then restrict our input set if we want to guarantee that the branch-and-bound procedure does not itself take exponential time.

---

[205] There are many general sorts of "relaxations" that we can apply to redistricting. For example, if we can formulate our redistricting problem as in integer-linear program, we might pretend that census blocks are infinitely divisible — converting this into a linear program that is simple to solve. From the solution to the linear program we might be able to compute bounds on our linear program. In general, however, choosing a successful relaxation of a problem requires mathematical intuition and cleverness.

Although we can make the automated redistricting process easier through restricting inputs, many useful restrictions may not be practical or possible. To eliminate the worst case behavior of these programs, we must enforce regularity upon the population blocks that we use, but many of these characteristics are exogenous. For example, geographical regularities can help us to draw compact districts. If your state is a perfect square with people uniformly distributed across it, then drawing equal-sized compact districts is simple. We might even require that all districts be equal-sized rectangles. Unfortunately, given the irregular boundaries and irregular distributions of population in many states, it is impossible to draw such regular districts, and finding the best set of districts is much harder. In sum, problems may be serendipitously easy, but we cannot rely on good luck.

Furthermore, we can only rely upon input restriction to eliminate worst-case behavior when we also carefully restrict our value functions. Restrictions that make the satisfaction of one goal easier may exacerbate the difficulties involved in satisfying other goals. There are few restrictions that have been shown to be useful across a variety of value-functions.[206] We may not be able to politically or normatively justify such restrictions.

---

[206] There is a notable exception. First, restricting the maximum value (population, size, etc.) that any one population block can be assigned to can sometimes allow problems to be solved in polynomial time. NP-complete problems that have been shown easily

Restrictions on Plans

Instead of trying to restrict goals or data, we might, more reasonably, restrict the types of districts or plans we will consider. For example, if we only consider districts that are contiguous rectangles, if our state has no "holes" (e.g., lakes are not counted as part of the district), and we are interested only in maximizing one form of compactness (in this case, perimeter-minimizing), we can more easily find "optimal districts" (Johnson 1982). Most political planners, however, would consider such restrictions unrealistic and overly restrictive.

We should also remember that while stating restrictions in this way is mathematically convenient, politically and normatively such restrictions imply a set of restrictions on value functions or inputs. For example, if we require that our plans be

---

solvable under this restriction are referred to as "pseudopolynomial." NP-complete problems that have been shown to remain intractable under this restriction are termed "strongly NP-complete."

A number of redistricting sub-problems, when further restricted to draw only *two* districts are "pseudopolynomial" in this sense. That is, if we are trying to divide an area into exactly two districts, and we can set an upper bound on the value of each population block, we can compute the optimal plan in polynomial time for many, but not all, sub-problems. These same problems are strongly NP-complete for more than two districts. See Appendix A for more details.

contiguous we exclude any value functions that allows contiguity to be weighed against other goals. Because of this equivalence, we cannot escape the limitations of value and input restrictions by reformulating them as plan-restrictions. Although a plan restrictions may be valuable for formulating issues clearly, we must be careful that these are not used by political manipulators to obscure the other types of restrictions that they imply.

5.4.4. Can Sub-Optimal Redistricting Help?

When we defined theoretical computational complexity, our solution concept was very demanding. Political solutions, however, seldom require the precision that we demand of theory. If we relax our requirements for precision, can we find general, easy, practical methods of redistricting? Can we find optimal plans most of the time, or find approximations that are "close enough"?

Optimal Redistricting — Most of the Time

There are two possible ways in which we could quickly perform optimal redistricting "most" of the time: We might develop a method to get close to optimum for all cases, or to find the optimum for most cases.

Remember that when we defined computational-complexity, we required that our algorithms be guaranteed to generate a *correct* solution to a problem. We might relax this guarantee of correctness, and allow our methods to make occasional errors.[207] The

---

[207] Technically, we expand our solutions to algorithms which produce the result with

possibility of "solving" some NP-complete problems has not been theoretically foreclosed. Unfortunately, it is conjectured that NP-complete problems are not susceptible to this type of solution[208] (Johnson 1984; Papadimitriou 1994).

Instead of relying on probabilistic methods, we might hope for an easy draw of problems — it might be that an algorithm solves the redistricting problem with certainty either on average or in practice.[209] As long as there exists some case for which

---

probability ε — we are not here bounding the magnitude of the error, only the probability of its occurrence.

[208] An algorithm is said to be "boundedly probabilistic polynomial" (BPP) if it computes a "solution" to all instance of a problem in polynomial time, and the "solution" computed has a probability strictly greater than 50% of being correct, each time the algorithm is executed. Because the probability of error on *each* execution of the algorithm is independent, algorithms in this class can be executed repeatedly to obtain any desired probability of correctness. The conjecture referred to above is that no NP-complete problem is a member of the class BPP.

[209] Technically, we might say that an algorithm works well *on average* when we describe its behavior upon problems drawn from a theoretical, mathematically-defined probability distribution; whereas, we would claim an algorithm works well *in practice* when we have evaluated it against real empirical data.

exponential time is required, the problem is still, by definition, computationally intractable, but it may not be quite easy most of the time.

This is primarily an empirical question: How well do real methods work on real data? As Section 2 showed, however, the track-record of automated redistricting is not very good. Researchers' previous attempts to generate optimal districting plans suggest that this problem can be quite difficult "in practice," and the literature reveals no procedure that is both demonstrably optimal and that has been generally effective on data sets large enough to be useful for political redistricting.[210] Yet, the question of whether redistricting sub-problems can be solved in practice remains open. Analysis of average-case complexity requires, in most cases, that we specify a particular distribution function. My literature search reveals no specific average-case complexity results concerning the redistricting sub-problems discussed above for any distributions.[211] Because of these, A formal analysis of average case properties of the redistricting problem is beyond the scope of this chapter.

Finally, a recent theoretical result should give us pause before we place our hopes on the average case: It has been proved that for the class of simple computable distributions,

---

[210] See Gudgin and Taylor (1979) for a discussion of some early attempts.

[211]It is not obvious what mathematical function can be used to accurately describe the distribution of values over population blocks (though see Gudgin and Taylor (1979)).

the average-case complexity of all NP-complete problems is exponential (Li and Vitanyi 1992). For some distributions, average-case complexity cannot be an escape from intractability.

As the previous section shows, the prospects of discovering a tractable procedure for optimal redistricting are dim. In practice, automated procedures will almost certainly result in sub-optimal redistricting. This section discusses both practical and political implications concerns surrounding approximation methods.

Guaranteed Approximations

Computational complexity is a measure of the difficulty of obtaining *optimal* solutions, but it says little about the difficulty of approximation. In fact, NP-complete problems, while equivalent for optimal solutions, are not equivalent for approximation. Some problems are much easier to approximate than others. For some NP-complete optimization problems there are methods that will, in polynomial time, generate a solution that is *guaranteed* to be within a particular percentage of the optimal value.[212]

---

[212] One must be careful to distinguish between measurements of approximation based on the value of the solutions produced versus measures based on percentage of solutions which are excluded. For example, simple hand drawn districts are likely to be in the top 99% of all the possible districts - simply because there are so many possible districts with little value. However, these simply drawn districts may be much lower in value than the

Methods that obtain arbitrarily close "solutions" in polynomial time are known as *fully polynomial approximations.* Unfortunately, it can be proved that no fully polynomial approximations exist for many of the redistricting sub-problems discussed above.[213]

Although the redistricting problem does not allow arbitrarily close approximations, we have not excluded the possibility of approximating a solution to within some fixed percentage of the optimum. No such guaranteed approximation procedure for a redistricting sub-problem has yet been demonstrated, but the question remains open.[214]

Making an Educated Guess

---

optimal district.

[213] Epsilon approximations are methods that guarantee that the ratio of the value of the approximal solution to the value of the true optimum is no less than 1-ε. Fully polynomial approximations are epsilon approximations which with time requirements bounded by a polynomial function of ε, for all ε.

It has been shown that no fully polynomial approximations exist for problems which are *strongly* NP-complete (Papadimitriou 1994). As Appendix 1 shows, a number of redistricting sub-problems are at least strongly NP-complete.

[214] I conjecture that good guaranteed approximation limits can be obtained for the problem of minimizing population inequalities between districts.

By far the most common way of approaching automated redistricting is through the use of *heuristics*. A heuristic is any methods that is considered useful for problem solving, but for which no guarantees of optimality (or approximate optimality) apply. While heuristic methods give no guarantee of approximate optimality, they can reduce the set of plans that need to be considered.

As I noted in Section 2, all of the procedures that researchers have used for automated redistricting are in fact heuristics. The computer science and operations research literature also offers several heuristics that have been useful in solving mathematically similar (partition) problems:

- Simulated annealing, a process that is analogous to the slow cooling in metals, has been suggested for use in the redistricting problem (Browdy 1990b). Zissimopoulos (1991) uses one variation of this technique on a number of set-partition problems.

- Genetic algorithms use processes analogous to crossover, mutation, and evolutionary selection to generate solutions to optimization problems. Chandrasekharam(1993) examines the performance of some variants of genetic algorithms on selected graph-partitioning problems.

- (Hershberger 1991) develops a polynomial time algorithm that divides a polygon into two optimally compact subregions. He suggests a divide-and-conquer heuristic for the more general polygon-partitioning problem, where his algorithm would be used to repeatedly subdivide a polygon.

•(El-Farzi and Mitra 1992) transforms set-partition problems into integer-programming problems *Lagrangean relaxation* and other assignment relaxation heuristics as part of a *branch-and-bound* solution method.

The heuristics above have been shown to be useful in several sets of test cases and merit further investigation for redistricting. However, the size of the test problems in the papers above corresponded approximately to redistricting problems on the order of 10 districts and 100 population units — much smaller than the typical, practical, redistricting problem. Furthermore, for at least one common sub-problem, drawing compact districts according to one common definition of compactness, many of these methods have performed poorly. (See Chapter 2 and Chapter 4.)

Political Implications of Sub-Optimal Redistricting

While sub-optimal redistricting methods offer the only practical means of implementing automated redistricting, these methods have a number of disturbing normative and political implications.

In his argument for automated redistricting, Isaccharoff writes (emphasis added):

> If reviewing courts use the absence of a *verifiable computer algorithm* or some other clear ex-ante articulation of the bases for reapportionment decisions as presumptive evidence of constitutional infirmity, the logic of adjudication may - as with the development of fixed equipopulation rule after *Reynolds* - propel the states toward the use of verifiable criteria for reapportionment (Issacharoff 1993).

All current methods for automated redistricting are heuristic—they offer no guarantee that a solution is close to the optimum. In practice, we can attempt to gauge the effectiveness of a heuristic process thorough experimentation and simulation, but such analyses will depend on the value functions we use. If we cannot place guarantee their performance, automated redistricting methods lose much of their normative appeal.

Furthermore, judicial review becomes more difficult, as the quality of the plans that we generated becomes a very technical question, dependent both on the choice of value functions used and heuristic methods. It will be difficult to determine whether a method is biased, and also difficult to determine whether a method is *narrowly tailored* to pursue legitimate goals. If, for example, creation of minority-opportunity districts is one of our goals, how will we determine that it has not been weighed too heavily by the redistricting algorithm?

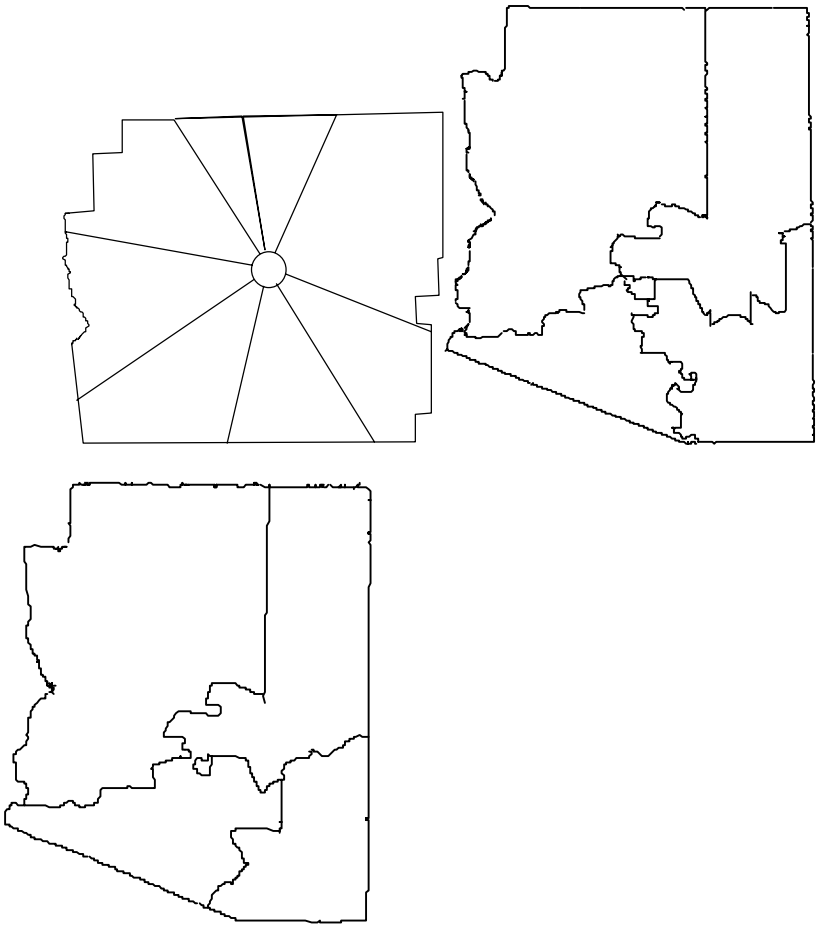
Second, guarantees of performance notwithstanding, heuristics and approximation techniques select from a large body of solutions, and may be biased in their selection. Even if we select only among methods which arrive at solutions in the top 99.99% of all possible district plans, the pool of possible solutions is staggeringly large, and the actual values of such solutions may differ dramatically. Solution methods can be biased toward certain classes of solutions.

Heuristics may leave plenty of room for manipulation. The *Ohio State University* (OSU) is a particularly clear example of a method that both embodies a particular bias and that can be manipulated for a variety of goals. The OSU method draws plans that are

arbitrary in their particulars but follow the same general plan: they make one large central district, and then other districts that radiate out from this central district (Hale 1966) (Figure 1a). In contrast to the OSU method's ideal, Arizona's 1980 redistricting plan used the same type of arrangement for a racial gerrymander (Figure 1b). The final court approved plan is also based upon the same general lines (Figure 1c) (Congressional Quarterly Staff 1983; Light 1982).[215]

---

[215] I have removed all but the state boundary and district lines from the maps in Figure 1 in order to maximize the clarity of the illustration.



A *B* *C*

Figure 1: Will The Real Gerrymander Please Stand Up?

Bias can also enter the process in the form of initial conditions from which these heuristics start. Since most redistricting heuristics are *path-dependent*, the starting conditions for these methods can influence the types of plans that they generate. For example, the two most popular automated redistricting algorithms, the Weaver-Hess procedure, and the Nagel method, both described in Section 2, are sensitive to initial conditions. As Weaver writes (Weaver 1970):

> "CROND's districting technique does work by trial and error...Different starting points, whether by hand or mechanical techniques discussed below, will usually produce different sequences and different sets of districts."

Unless all district plans near the optimum are very similar in the *types* of plans they produce, how we choose starting points matters. These choices offer opportunities for hidden political manipulation of the outcomes. While individuals may not be able to use such manipulation to choose a precise plan, interest groups may be able to use methods that are biased toward those groups. Perversely, this could make partisan gerrymandering easier, because it would limit the gains that incumbents could make on their own — giving incumbents in the same party a stronger incentive to cooperate.

The intertwining effects of value functions, methods, and starting points weaken the distinction between intent and effect that the automation procedure is supposed to make clear. The veil of automation has been pierced. Judicial review of such intertwined and technical procedures may be significantly more difficult than judicial review of the individual districting plans. Furthermore, political groups with large computing resources will be much better equipped to discover the combinations of methods and functions that will work to their advantage.

If manipulation cannot be eliminated by automation, perhaps the opportunities to choose plans can be opened up equally to all parties. Why not make computer resources

available to all parties, let interested parties submit plans, and select the plan with the highest value?

This scheme has two shortcomings. First, the incentives remain for strategic manipulation at the value-choice stage. Automation cannot be used as a "veil of ignorance" to force participants to bargain fairly. Second, this scheme discourages compromise. Since under this scheme the plan with the highest value function acts as a "trump" and is implemented *in toto*, there is a strong incentive for secrecy. If an opponent knows the details of your proposed plan, he will be able to know whether more work needs to be done on his plan. In addition, he may be able to use iterative improvements to submit a slightly higher valued plan that better suits his goals.

5.5. Automating the Redistricting Process Limits Conflicts with Representational Goals

Even if we manage to overcome the technical difficulties involved in finding optimal districts, we will still need to formalize the notion of an "optimal district." In attempting this, we are bound to encounter two political and normative problems. First, technical details of our formulation may be used to manipulate the redistricting process — we will be able to manipulate by defining our goals in particular ways. Although automated redistricting may limit incumbent gerrymandering, it is likely to encourage partisan and interest-group gerrymandering. Second, the very act of formalizing our goals in a form suitable for computer processing will eliminate from consideration a number of important representational goals.

5.5.1. Limit on the Formalization of Redistricting Goals

To automate the redistricting process, the goals we use must first be quantified — the extent to which a plan meets (or does not meet) each of the goals must be representable by a number.[216] For example, population inequality might be quantified by subtracting the largest and smallest districts.

Not all goals may be easily quantified, however. In particular, many goals are *representational* – they refer not to a directly measurable aspect of a district, such as its population or shape, but to the role that district plays in creating political fairness. As Justice White wrote for the court in *Gaffney* v. *Cummings* (1973):

> The very essence of districting is to produce a different — a more "politically fair" — result than would be reached with elections at large, in which the winning party would take 100% of the legislative seats.

Representational goals, such as protection of communities of interest and nondilution of minority representation, are especially difficult to describe formally.

The Court has recognized a wide variety of valid goals for redistricting in a number of cases. In *Karcher vs. Daggett* (1983), the court's opinion stated that "any number of

---

[216] In theory this number might represent either an ordinal or cardinal ranking. In practice, all of the automated procedures described in the literature have used cardinal rankings.

consistently applied legislative policies" might be important enough to justify deviations from absolute population equality, including compactness, protection of municipal boundaries, preserving the core of previous districts, and protecting incumbents. In *Davis* v. *Bandemer* (1986), the court determined that "vote dilution" is a relevant and justiciable factor in redistricting. Most recently, in *Miller* v. *Johnson* (1995), the court chastised the defendents for ignoring "traditional" principles, including (but not limited to) contiguity, compactness, "respect for political subdivisions or communities defined by actual shared interests," and race neutrality[217]. Could we program a computer to evaluate and balance all of these factors?

---

[217] The complexity and subtlety of representational goals is also illustrated by the court's commentary on multi-member districts in *White* v. *Regester* (1973), and by the similar commentary found in the Senate report on 1982 amendments to Section 2 of the Voting Right's Act.

The high court, in *White* v. *Regester* recognized that the "totality of the circumstances" must be used in the evaluation process. While the court listed a number of important circumstantial factors including racial motivation for a plan, history of discrimination and voting patterns, and the number of majority-minority districts, it stressed that decisions were not limited to these factors.

After listing the seven well known "totality of the circumstances" factors, the Senate committee states:

To evaluate plans with respect to representational goals, we need to be able to recognize natural geographic, social, and historical patterns, and to use extensive knowledge about political and social relationships.[218] It is exactly such patterns that computers are least able to recognize, especially when these patterns are dynamic, and cannot be simply "hard-wired." In fact, humans perform better than any computer

---

> While these enumerated factors will often be the most relevant ones, in some cases other factors will be indicative of the alleged dilutions.
>
> The cases demonstrate, and the Committee intends that there is no requirement that *any particular number of factors be proved, or that a majority of them point one way or another* (U.S. Congress. Senate. Judiciary Committee 1982, emphasis added).

Although these factors were to be used when evaluating whether multi-member at-large districts were discriminatory and neither the court nor the Senate report specified that these factors should be used when drawing district lines, they show how complex the concept of fair representation can be.

[218] Even compactness in some of its forms is difficult to formalize. For example, Grofman (1993) argues that instead of using purely formal compactness criteria for evaluating district geography, we should turn to *cognizability*, the ability of a legislator to define, in commonsense language, based on geography, the characteristics of her district.

algorithms in just these sorts of *natural* recognition tasks (Dreyfus 1992). Evaluation of any one of the circumstantial factors listed above is beyond the reach of current computer technology, and it is unlikely that the "totality of the circumstances"[219] can be adequately formalized in the foreseeable future.[220] The legislature would be abdicating its responsibility if it allowed such goals to be jettisoned in favor of automation.

Even goals that may seem to be simple, easily quantified, and even generally agreed upon may be subject to several different, and unequal, quantifications. For example Niemi, et al. (1991) lists close to two dozen quantifications of the goal "compactness," most of which will differ (in at least some cases) in the values they assign to districts. For example, the controversy over apportionment methods (how to allocate representatives on the basis of population) in the U.S. shows that even seemingly small differences in the formalization of an informal standard may be of great practical importance.[221]

---

[219] Although it might be claimed that the three prongs of *Thornburg* v. *Gingles* has eliminated the need to look at the totality of the circumstances, this is not the case. *Growe* v. *Emison* and *Voinovich* v. *Quilter* and *Johnson* v. *DeGrandy* clearly assert that these three conditions are necessary but not sufficient.

[220] See also Karlan (1993), for further arguments concerning the limited ability of formalism to capture representational goals.

[221] See Balinski and Young's excellent book on the politics of choosing an

Furthermore, redistricting typically involves weight multiple social goals.[222] The number of possible weights and weighting functions is literally infinite. Even should

---

apportionment method (Balinski and Young 1982). Also see *U.S. Department of Commerce* v. *Montana* (1992) for an overview of this history and its legal significance.

[222] Many state constitutions contain multiple criteria for evaluating districts (Grofman 1985). An extreme example of what an automated procedure would have to accomplish is illustrated by the Hawaiian constitution, which requires the following:

a. No district shall extend beyond the boundaries of any basic island unit.

b. No district shall be so drawn as to unduly favor a person or political faction.

c. Except in the case of districts encompassing more than one island, districts shall be contiguous.

d. Insofar as practical, districts shall be compact.

e. Where possible district lines shall follow permanent and easily recognized features, such as streets, streams, and clear geographical features, and when practicable shall coincide with census boundaries.

f. Where practicable, representative districts shall be wholly included within senatorial districts.

overall redistricting goals and their formulations be agreed upon, the choice of weights may still be subject to contention. The complexity of weighing different goals should not be underestimated. It is unreasonable to expect that a dynamic and subtle weighing of social values can be captured by a simple fixed set of linear weights. A study by Sheth & Hess in which political scientists were asked to give fixed linear weights to two popular redistricting goals (Sheth and Hess 1971) provides an example of the dramatic differences in weighting that can occur even in a relatively homogenous group deciding among extremely restricted options.

5.5.2. Political Implications of the Formalization of Redistricting Goals

You might well argue that while these three limits on formalization should not be ignored, they are not limits on formalization *per se*, but only limits on the interpretation of a particular formalization as "objectively neutral." Some proponents of automated redistricting (Browdy 1990a; Issacharoff 1993; Vickrey 1961) do not claim that automation makes redistricting neutral, but instead specify that the decisions involved in choosing and formalizing redistricting goals must be made by a political process. Even

---

g. Not more than four members shall be elected from any district.

h. Where practicable, submergence of an area in a larger district wherein substantially different socio-economic interests predominate shall be avoided.

when the goals and formalizations are chosen by a political process, however, we should expect that this type of formalization will have its disadvantages.

Seemingly simple characterizations of formal criteria, such as geographical compactness, can have counterintuitive implications for individual cases.[223] Given this fact, and the multiplicity of possible formalizations that we can develop even for such seemingly intuitive goals as "geographic compactness," automated redistricting will have three serious defects:

• First, automation will shift the political debate from redistricting goals to technical details because there are so many differing characterizations of common goals each with potentially different political effects. Automation therefore seems likely to engender extensive and arcane battles over the mathematical details of goal functions. As Richard Nelson writes in *The Moon and the Ghetto* (Nelson 1977): "However, there surely is a tension between the language of optimization... and creative problem solving, which implicitly is understood to be the proper language for looking at the real decision process."

• Second, requiring that all redistricting criteria be formalized will disadvantage social values that are difficult to mathematize. Representational goals such as preserving communities of interest or preventing minority vote dilution are particularly difficult to quantify because they succinctly condense a host of dynamically changeable, interacting

---

[223] See Young (1988) and Chapter 2 for examples of nonintuitive districts resulting from seemingly simple compactness measurements.

social factors. Current sophistication in computer programming is simply insufficient to capture such goals.

• Proponents of automated redistricting claim that it eases judicial and public review by making political purposes clear. Instead, formal characterizations often mask, rather that clarify, political purposes. In cases where formal criteria have counterintuitive implications, debate over particular district plans may illuminate political motives much more clearly than debate over the technical characterizations of redistricting goals.[224]

5.6. Discussion

Computers are eminently useful tools for many purposes — including redistricting. Unfortunately, good tools cannot always eliminate political and social problems: Automated redistricting is neither as simple nor as objective as has been claimed.

In this chapter I have demonstrated that the general redistricting problem and common sub-problems belong to a class of problems that is widely believed to be computationally intractable. Practical methods for generating districts automatically will almost certainly be based on heuristics— procedures which make guesses at solutions.

---

[224] Since the consequences of particular formulations are often less clear than the consequences of plans, one might expect that *risk averse* bargainers would prefer to bargain over plans than to bargain over goals, *ceteris paribus*. The ceteris *paribus* assumption is a strong one, however, and a formal game model of the bargaining process is needed to investigate these implications further.

The current proposals for implementing automated redistricting make exactly such guesses. This guesswork is disturbing because researchers offer little in the way of empirical data or theoretical analysis to show that these methods are successful in implementing our goals, and objective in excluding other considerations.

Even if we develop heuristics with better empirical and theoretical support, or go beyond heuristics altogether, we will run into political and normative difficulties when we try to make every districting goal recognizable to a computer algorithm. Representational goals, which are based upon the analysis of social and political patterns, are well beyond the current state of computer technology to quantify. Many simpler goals, such as geographic compactness, can be easily quantified, but are open to a plethora of conflicting quantifications.

If districts are created in an open political process, we may require only that districts meet minimum standards, such as population equality, in order to provide a "fair playing field"; and we may let the political procss determine the rest. If, instead, we override the political process and substitute for it a computer program, we must have confidence that the computer can not only satisfy our minimum standards, but that it can affirmatively meet our representational goals. There is a vast difference between the two.

Proponents claim that automated redistricting promotes fairness and illuminates the redistricting process. They assume that social goals for redistricting are easily and simply characterized. In fact, as this chapter argues, even the most seemingly straightforward of redistricting goals may be subject to many conflicting and confusing technical

characterizations. More subtle social goals, especially those goals which are explicitly representational, may be given short shrift or have to be disregarded altogether to accommodate the automation process. Thus, the automation process advantages nonrepresentational goals over representatational one — a telling vice for an attempt to design a system of representation.

Some proponents also claim that automated redistricting can operate neutrally — merely implementing social goals chosen in a previous political debate. In fact, representational values are difficult or impossible to adequately formalize using current techniques, and no feasible methods for finding optimal plans for arbitrarily specified value functions have been discovered. Heuristic methods further complicate this picture because they intertwine our choice of values with how we create districts.

Proponents claim that automated redistricting eases judicial review. In fact, the technical complexities of characterizing goals formally, and of finding plans to serve these goals, is a barrier to judicial and public review and to participation in the districting process.

Finally, proponents have claimed that redistricting will reduce political manipulation. In fact, automated redistricting may only serve to change the types political manipulation that occurs. Automation may reduce political manipulation that uses redistricting plans for the purpose of advantaging particular individuals. It may promote, however, political manipulation using formal characterization of goals and

choice of automation methods so as to advantage particular political groups. It may disadvantage particular politicians, but strengthen partisanship.

Proponents hold up automation as a veil of ignorance that can be used to promote fairness and prevent manipulation in the redistricting process. Unfortunately, automation may not eliminate the opportunity to politically manipulate, but instead shift that opportunity toward those groups that have access to the most extensive computing facilities. At the same time, automation shrouds this manipulation under the illusion of neutrality. Even if resources are equal, the potential for political side effects stemming from the choice of different characterizations of values and different heuristics for implementing plans, pierces the veil of ignorance and subverts the fairness of the automated process.

Appendix: Proofs

The most common way to show that a problem is NP-complete is to show that it can be polynomially reduced to another problem that is already known to be NP-complete. Similarly, the most common way to show that a problem is NP-hard is to show that solving it requires solving a problem that is known to be NP-complete. I will use these two general methods to show that the redistricting sub-problems discussed in Section 3.5 are NP-hard.

First, we will need some notation:

- We will use $\mathbf{x}_i$ to refer to the *i*th census block. These blocks are vector-valued, and we will assume that blocks comprise population values, partisan registration percentages, and geographic locations, among other features.
- We will use $\mathbf{d}_i$ to refer to the *i*th district. A district is a set of census blocks: $\mathbf{d}_i = \{\mathbf{x}_j, \mathbf{x}_k, \ldots, \mathbf{x}_n\}$.
- We will use $\mathbf{p}_i$ to refer to a particular plan. A plan is a partition, which has been defined in the Section 3.1 (footnote 20), of the set of all census blocks into a set of districts: $\mathbf{p}_i = \{\mathbf{d}_1, \ldots, \mathbf{d}_n\}$.

We can now move to the proofs. As the proofs are repetitive, I will show the first in more detail, and sketch the remaining three.

Claim 1: *Creating Equal Population Districts is NP-Hard*

Definitions: First we need to define a measure of population inequality. We can compute the population in a district by summing over its census blocks:[225] $\mathrm{pop}(\mathbf{d}_i) = \sum_{\mathbf{x}_i \in \mathbf{d}_i} \mathrm{pop}(\mathbf{x}_i)$. We will use a simple measure of the inequality of a plan, $V(\mathbf{p}_i)$, the difference between the population of its largest and its smallest districts:[226] $V(\mathbf{p}_i) = \sup_{\mathbf{d}_i \in \mathbf{p}_i} \mathrm{pop}(\mathbf{d}_i) - \inf_{\mathbf{d}_i \in \mathbf{p}_i} \mathrm{pop}(\mathbf{d}_i)$.

Proof: Given a particular set of census blocks that we must district, let us label the set of all possible plans **P**. To draw districts that are as close as possible to this goal, we must find a plan, $\mathbf{p}^*$ that minimizes *V* over all possible plans using those census blocks, and having a fixed number of districts, *k*:

$$\mathbf{p}^* = \arg\inf_{\{\mathbf{p}_i \in \mathbf{P} \mid |\mathbf{p}_i| = k\}} V(\mathbf{p}_i).$$

Suppose we produce an algorithm that finds $\mathbf{p}^*$, for any set of census blocks. Given $\mathbf{p}^*$ it is trivial to decide the question: "Is there a plan such that $V(\mathbf{p}^*)=0$?"

---

[225] I assume that the population in each census block is fixed and positive.

[226] The redistricting problem remains NP-hard as long as $V(\mathbf{p}_i)$ equals 0 if and only if we have absolute population equality — my particular measurement of population equality is only for convenience.

But if we can answer this question we can solve an NP-complete problem. For *k*=2, answering the question is equivalent to solving the following NP-complete problem (listed in (Garey and Johnson 1983)), simply relabel our census blocks *a*, relabel our two plans *A′* and (*A*-*A′*), and relabel our population meaurement *s*:[227]

Set Partition

Instance: Finite set *A* and a size $s(a) \in Z^+$ for each $a \in A$.

Problem: Is there a subset $A' \subseteq A$ such that $\sum_{a \in A'} s(a) = \sum_{a \in A - A'} s(a)$?

This problem is NP-complete, but not strongly NP-complete. If we need to create a plan with more districts, the problem is strongly NP-complete. If we can decide whether V($\mathbf{p}^*$)=0 for *k*>2 then we can answer the following strongly NP-complete question (listed in (Garey and Johnson 1983)), by performing a similar relabeling:

3-Partition

---

[227] For an alternative formulation of partitioning as an integer program, see El-Farzi and Mitra (1992).

Instance:[228] Set A of 3m elements, a bound $B \in Z^+$ and a size $s(a) \in Z^+$ for each $a \in A$ such that $B/4 < s(a) < B/2$ and such that $\sum_{a \in A} s(a) = mB$.

Question: Can A be partitioned into m disjoint subsets such that for $1 \leq i \leq m$, $\sum_{a \in A_i} s(a) = B$?

Claim 2: *Creating A Maximally Compact District Plan is NP-Hard*

Definition: We will measure the compactness of a district as maximum distance between the centers of any pair of census blocks in that districts,[229] and we will measure the compactness of a plan by measuring its least compact district.[230]

---

[228] Since the 3-Partition problem does assume more restrictions for its set-members than we put on our census blocks, this shows our redistricting problem to be NP-Hard, but is not sufficient to show that it is NP-complete.

[229] Using this measurement, large values of V indicate the district is ill-compact. Papayanopoulos (1973) uses a similar definition of compactness. Furthermore, the maximum distance between census blocks in a plan is used as part of calculating a large number of other indices, see Niemi et al. (1991).

[230] It is a common practice to use the worst district to measure the compactness of an entire plan (Papayanopoulos 1973).

$$V(\mathbf{d}) = \sup_{\mathbf{x}_i, \mathbf{x}_j \in \mathbf{d}} \left(\text{distance}\left(\mathbf{x}_i, \mathbf{x}_j\right)\right)$$

$$V(\mathbf{p}) = \sup_{\mathbf{d}_i, \mathbf{d}_j \in \mathbf{p}} \left(V\left(\mathbf{d}_i, \mathbf{d}_j\right)\right)$$

Proof: Again, suppose we produce an algorithm that finds a $\mathbf{p}^*$ that solves our problem for any set of census blocks. Given $\mathbf{p}^*$ it is then trivial to decide the question (for some given constant *D*): "Is there a plan such that $V(\mathbf{p}^*)<D$?"

But if we can answer this question, we can solve an NP-complete problem. Answering the question is equivalent to solving the following problem, which is (strongly) NP-complete for any $k>2$ (discussed in (Johnson 1982)):

Distance-d Partition of Points in the Plane

Instance: Finite set $\mathbf{P} \subset Z \times Z$ of integer-coordinate points in the plane, positive integers *D* and *k*.

Problem: Is there a partition $\mathbf{P} = \mathbf{P}_1 \cup \ldots \cup \mathbf{P}_k$ such that if *u* and *v* are distinct points in some set P_i of the partition, then the Euclidean distance $d(u,v)<D$?

Claim 3: *Creating A Plan with Maximally Competitive Districts is NP-Hard*

Definition: Define a maximally competitive plan as one that minimizes the overall expected difference between Republican (R) and Democratic (D) registration[231] in each district:

---

[231] Instead of registration, we can use expected Republican and Democratic turnout, as long as the expected turnout in a census block is independent of the expected turnout in

$$V(\mathbf{d}) = \sum_{\mathbf{x} \in \mathbf{d}} R(\mathbf{x}) - D(\mathbf{x})$$

$$V(\mathbf{p}) = \sum_{\mathbf{d} \in \mathbf{p}} (V(\mathbf{d}))^2$$

Proof: Again, suppose we produce an algorithm that finds a $\mathbf{p}^*$ that solves our problem for any set of census blocks. Given $\mathbf{p}^*$ it is then trivial to decide the question (for some given constant *D*): "Is there a plan such that $V(\mathbf{p}^*) < D$?"

Again, if we can answer this question, we can solve an NP-complete problem. Answering the question is equivalent to solving the following problem, which is NP-complete (discussed in (Garey and Johnson 1983)):

Minimum Sum of Squares

Instance:[232] Finite set A, a size $s(a) \in Z^+$ for each *a*, positive integers $k \leq |A|$ and *D*.

Problem: Can *A* be partitioned into *k* disjoint subsets such that $\sum_{i=1}^{k} \left(\sum_{a \in A_i} s(a) \right)^2 \leq D$?

---

the district.

[232] Note that the minimum sum of squares problem assumes that each set member is positively valued. This makes it a subset of the competitive redistricting problem, because Democrats may outnumber Republicans in a census block. Hence this demonstration shows that the redistrict is NP-hard but is not sufficient to show that it is NP-complete.

This problem is NP-complete, but is not strongly NP-complete for fixed k (hence this is strongly NP-complete in the number of districts not in the number of census blocks). NP-completeness is preserved if the exponent 2 is replaced by any fixed rational number $\alpha>1$.

Claim 4: *Creating A Contiguous Equal-Population Redistricting Plan is NP-Hard*

Definition: We will use the same measurement of population as we did in the first problem. In addition, if any two census blocks i,j are contiguous, we will connect them with a unique edge $e_{i,j}$. Our problem in this case is to find a partition into k districts, such that the edges fully contained in that district form a connected graph, and such that we minimize $V(\boldsymbol{p})$, as in problem 1.

Proof: Again, suppose we produce an algorithm that finds a $\mathbf{p}^*$ that solves our problem for any set of census blocks. Given $\mathbf{p}^*$ it is then trivial to decide the question (for some given constant D): "Is there a plan such that all districts are contiguous and $V(\mathbf{p}^*)<D$?"

If we can answer this question, we can solve an NP-complete problem. Answering the question is equivalent to solving the following problem, which is (strongly) NP-complete (discussed in (Johnson 1982; Johnson 1984)):

Cut into Connected Components of Bounded Weight

Instance:[233] Graph **G**=(**V,***E*), a size $s(v) \in Z^+$ for each *a*.

Problem: Is there a partition of **V** into disjoint sets $\mathbf{V}_1$ and $\mathbf{V}_2$ such that $\sum_{v \in \mathbf{V}_1} s(v) \leq K$ and $\sum_{v \in V_2} s(v) \leq K$ and both $\mathbf{V}_1$ and $\mathbf{V}_2$ induce connected subgraphs of **G***?*

---

[233] Note that the minimum sum of squares problem assumes that each set member is positively valued. This makes it a subset of the competitive redistricting problem, because Democrats may outnumber Republicans in a census block. Hence this demonstration shows that the redistrict is NP-hard but is not sufficient to show that it is NP-complete.