MIT Libraries | DSpace@MIT

MIT Open Access Articles

Modeling tax evasion with genetic algorithms

The MIT Faculty has made this article openly available. ***Please share*** how this access benefits you. Your story matters.

Citation: Warner, Geoffrey, Sanith Wijesinghe, Uma Marques, Osama Badar, Jacob Rosen, Erik Hemberg, and Una-May O'Reilly. "Modeling Tax Evasion with Genetic Algorithms." Econ Gov 16, no. 2 (November 18, 2014): 165-178.

As Published: http://dx.doi.org/10.1007/s10101-014-0152-7

Publisher: Springer Berlin Heidelberg

Persistent URL: http://hdl.handle.net/1721.1/104640

Version: Author's final manuscript: final author's manuscript post peer review, without publisher's formatting or copy editing

Terms of Use: Article is made available in accordance with the publisher's policy and may be subject to US copyright law. Please refer to the publisher's site for terms of use.

Massachusetts Institute of Technology

Econ Gov (2015) 16:165–178
DOI 10.1007/s10101-014-0152-7

ORIGINAL PAPER

Modeling tax evasion with genetic algorithms

Geoffrey Warner · Sanith Wijesinghe · Uma Marques · Osama Badar · Jacob Rosen · Erik Hemberg · Una-May O'Reilly

Received: 29 November 2013 / Accepted: 8 November 2014 / Published online: 18 November 2014
© Springer-Verlag Berlin Heidelberg 2014

Abstract The U.S. tax gap is estimated to exceed $450 billion, most of which arises from non-compliance on the part of individual taxpayers (GAO 2012; IRS 2006). Much is hidden in innovative tax shelters combining multiple business structures such as partnerships, trusts, and S-corporations into complex transaction networks designed to reduce and obscure the true tax liabilities of their individual shareholders. One known gambit employed by these shelters is to offset real gains in one part of a portfolio by creating artificial capital losses elsewhere through the mechanism of "inflated basis" (TaxAnalysts 2005), a process made easier by the relatively flexible set of rules surrounding "pass-through" entities such as partnerships (IRS 2009). The ability to anticipate the likely forms of emerging evasion schemes would help auditors develop more efficient methods of reducing the tax gap. To this end, we have developed a prototype evolutionary algorithm designed to generate potential schemes of the inflated basis type described above. The algorithm takes as inputs a collection of asset types and tax entities, together with a rule-set governing asset exchanges between these entities. The schemes produced by the algorithm consist of sequences of transactions within an ownership network of tax entities. Schemes are ranked according to a "fitness function" (Goldberg in Genetic algorithms in search, optimization, and machine learning. Addison-Wesley, Boston, 1989); the very best schemes are those

G. Warner · S. Wijesinghe (✉) · U. Marques
The MITRE Corporation, 7515 Colshire Drive, McLean, VA 22102, USA
e-mail: sanith@mitre.org

G. Warner
e-mail: gwarner@mitre.org

O. Badar · J. Rosen · E. Hemberg · U.-M. O'Reilly
Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology,
77 Mass. Ave, Cambridge, MA 02139, USA

that afford the highest reduction in tax liability while incurring the lowest expected penalty.

Keywords Tax evasion · Genetic algorithms · Agent-based modeling

JEL classification K340 · C630 · C730

1 Background and introduction

The U.S. tax gap, defined as the aggregate sum of the difference between the true tax liability and what is paid on time by all taxable entities, has recently been estimated to exceed $450 billion annually. The bulk of this difference (approximately 2/3) is attributable to individual taxpayer non-compliance, some of which originates from understated income hidden in "innovative" tax shelters comprising complex transactions among multiple business entities (GAO 2012; IRS 2006). Partnerships and other so-called "pass-through" entities are known to play a disproportionate role in these structures due to the relative flexibility of the tax rules governing transactions to which they are a party; in fact, the GAO estimates that at least $91 billion of the gap associated with individual non-compliance comes from this sector alone (GAO 2014).

Tax shelters are marketed to high net worth individuals by promoters. Promoters include banks, accounting firms, investment boutiques, and law firms that scour the tax code looking for exploitable loopholes. They then arrange and execute a sequence of transactions designed to reduce their client's tax liability. On the surface these transactions satisfy all relevant tax laws; on occasion, however, it becomes apparent that the transactions in question can have had no other purpose than the reduction of tax liability. Schemes[1] of this type have long been disallowed under a common law doctrine requiring that the associated transactions have "economic substance" (Robertson et al. 2010). They are now explicitly illegal under the provisions of the 2010 Affordable Care Act (IRS 2011).

These schemes come in a variety of shapes and sizes. Here we focus on an important subclass that rely on a mechanism called "inflated basis". "Basis" is the set point from which gains or losses are assessed for tax purposes; usually the basis of an asset is just the cost of acquiring it. There are, however, a complex set of rules governing how basis is computed or otherwise adjusted in the course of different transactions. By arranging for a sequence of these transactions among a commonly owned network of entities it is sometimes possible to enhance the basis of certain assets and thereby create artificial losses when these assets are ultimately sold. Such losses can then be used to offset gains elsewhere in a portfolio, thus reducing overall tax liability.

The sheer complexity of a typical tax shelter poses significant challenges to government enforcement efforts. Audits have traditionally been directed at single financial entities, whereas the most advanced schemes are conducted within vast networks of

[1] By "scheme" we shall mean a sequence of transactions arranged for the purposes of illegal tax evasion. It is worth noting, however, that in some instances the distinction between tax avoidance (which is legal) and tax evasion is not always clear.

these entities. Teasing out the detailed transaction flow associated with a particular scheme can therefore be prohibitively expensive or difficult, especially when that scheme is a new or previously unknown variant. The growing number and prevalence of pass-through entities such as partnerships has made this problem increasingly urgent (GAO 2010). An algorithm like the one we are proposing here would go a long way towards mitigating this problem by providing auditors with exemplars of potentially suspicious patterns of activity.

2 Approach and methodology

Allingham and Sandmo (1972) pioneered the application of microeconomic theory to the problem of tax compliance. Since then, the field has grown into a very active area of research involving practitioners from across a wide array of disciplines. Recent years have seen the introduction of computational methods like agent based modeling to the problem domain; these methods go beyond standard microeconomic models by treating individual taxpayers as discrete interacting entities embedded in complex social networks (Bloomquist 2006). Some notable examples of this kind of work include studies of changes in taxpayer reporting behavior under different audit scenarios (Hokamp and Pickhardt 2010), investigations of the impact of taxpayer network structures on tax compliance (Andrei et al. 2013) and the identification of critical audit rates at which whole populations suddenly shift their equilibrium level of compliance (Davis et al. 2003). In the present study, we propose an alternative formulation of the problem that treats tax evasion schemes, rather than individual taxpayers, as the fundamental objects of interest.

Tax evasion schemes are constantly changing. Whenever one is uncovered and measures are taken to eliminate it, others spring up to take its place. These others are often variations of the same underlying idea, though the flow of assets and the arrangement of involved entities may appear quite different than in the original scheme. One notable example of this phenomenon is the so called Son of BOSS tax shelter, which emerged in the mid-90s after its immediate predecessor, a strategy known as "shorting against the box", was rendered defunct by changes in the tax code (Wright 2013).

While there does not, as yet, exist a systematic method to anticipate the emergence of these schemes, they clearly share a basic formal structure. In fact we hypothesize that all such schemes are ultimately reducible to sequences of pairwise transactions between distinct financial entities. As these transactions are themselves governed by a finite set of rules, it is plausible to suppose that a computational model capable of generating candidate schemes automatically could be devised. We propose that a properly designed genetic algorithm (GA) is just such a model.

GAs are search heuristics, like hill climbing or simulated annealing, that can be applied to optimization problems. What distinguishes them from other search methods is their formal similarity to Darwinian evolution; the search process itself is mediated by a population of bit strings, or "chromosomes", which map to elements of the search space and can be manipulated in a manner reminiscent of their biological counterparts.

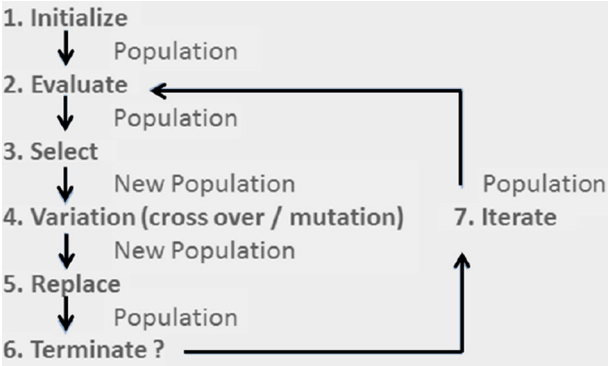

Fig. 1 Diagram of the the canonical genetic algorithm

We here undertake a brief overview of genetic algorithms and then explain our application of this methodology to the problem at hand.

2.1 Genetic algorithms

All GAs require a genetic representation (Goldberg 1989). This is a method for encoding solutions in a basic mathematical structure like a bit string or parse tree. For the sake of simplicity, we focus here on bit strings of fixed length K, which form the "chromosomes" of the representation. The representation itself consists of these chromosomes (the genotype), together with a deterministic mapping from each chromosome to an element of the search space (the phenotype). Further, all GAs require a measure of fitness on these phenotypes—loosely speaking, this constitutes the objective function of the problem at hand. The only formal requirement of the method of fitness evaluation is that it allow for an ordinal ranking of solutions. Finally, all GAs feature some method of selection and genetic variation. Selection involves choosing chromosomes in the population for reproduction according to the relative fitness of each member of the population—the higher the fitness, the higher the probability of being selected. Variation is introduced through the use of genetic operators like crossover and mutation. Crossover is the process whereby corresponding segments of two different chromosomes are chosen at random by some method and then transposed. Mutation consists of a bitwise flip of each element of the selected bit string with some probability p. We expand somewhat on these capsule definitions in what follows.

The canonical GA exhibits the following iterative structure, which we here describe in seven steps (Brabazon and O'Neill 2010). These steps are depicted in Fig. 1. First, an initial population of N chromosomes is generated, usually randomly. Second, each member of this population is subjected to an evaluation of its fitness. The process of fitness evaluation may be so simple it needs only the computation of a basic formula, as in the traveling salesman problem, or so complex that it requires its own simulation, as in the design of a bridge or a jet engine. In the third step, pairs of chromosomes are selected from the population for crossover and mutation. The selection method must favor the fitter members of the population; one popular approach, and the one

we adopt here, is called tournament selection. In tournament selection, k members are drawn at random from the population, and the fittest of these is selected. Fourth, the crossover and mutation operators are applied to the selected pair. Steps three and four are repeated until $N - e$ children have been produced, where e is the size of the elite population (that is, the group composed of the e fittest members of our original population). Fifth, the old population is replaced by the new; this latter is composed of the $N - e$ children that were produced by iterating steps 3 and 4, together with the e fittest members of the old population. Sixth, a test condition is evaluated to determine whether to halt. Seventh, assuming the test condition is false, we return to step 2 with our new population.

2.2 An application of genetic algorithms to tax evasion

Figure 2 illustrates the major functional components used in the GA methodology. For the purpose of representing a tax evasion scheme our algorithm begins by instantiating a number of Asset and Entity objects. The Asset class may include stocks, promissory notes, loans, cash, options or whatever other securities or instruments are deemed necessary. Entities include individual taxpayers, partnerships, trusts, corporations, banks, etc. These objects track all the book-keeping associated with transactions, including ownership, the market value of assets, the basis of assets, debt obligations, and tax owed on any particular exchange.

Next, we devise a transaction plan that details the sequence of Asset flows between Entities. For this purpose we employ a variant of the genetic algorithm approach known as grammatical evolution (GE) (Brabazon and O'Neill 2010). The principal difference between this method and other similar algorithms is in the genetic representation. In GE, chromosomes consist of lists of integers (called "codons"), whereas phenotypes are lists of executable instructions. The mapping from genotype to phenotype proceeds by means of a context-free grammar. A grammar consists of two sets of symbols, called "terminal" and "non-terminal", together with a set of overwrite or production rules. The non-terminal set always includes a "start" symbol. Production rules prescribe how particular non-terminal symbols are to be replaced by combinations of terminal and non-terminal symbols. In our case, the output of production rules results in a list of

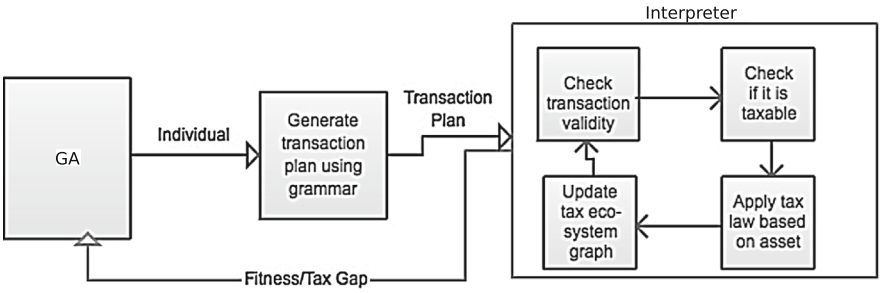

Fig. 2 Overview of tax simulator

Fig. 3 Example of a mapping from genotype to phenotype through a grammar. An integer string is converted into a transaction between the entities 'NewCo' and 'Brown' in which Brown gives Cash to NewCo in exchange for a Hotel

executable instructions (or "schemes") which act on previously instantiated Asset and Entity objects to produce a sequence of transactions.

Whenever a particular chromosome is being evaluated for fitness, a parser looks at the leftmost non-terminal symbol and replaces it using a production rule determined by the value of the corresponding codon. When only terminal symbols are left, they form a list of instructions which is then passed to an interpreter module for execution. The interpreter uses the Asset and Entity objects to carry out the instructions, most of which involve pairwise exchanges of Asset objects between Entities, and makes sure to apply whatever tax rules may be applicable to the exchange.

A simple example of our genetic representation is depicted in Fig. 3. Here the genotype consists of an arbitrary input string `[3, 11, 10, 5, 40, 7]`, which is mapped to the output function

$$\texttt{Transaction}(\texttt{NewCo}, \texttt{Brown}, \texttt{Material}(\texttt{200}, \texttt{Hotel}, \texttt{1}), \texttt{Cash}(\texttt{200})).$$

This output function instructs the interpreter module to remove a single Hotel asset worth $200 from the inventory of an entity called "NewCo" and to place it in the

inventory of an entity called "Brown", while simultaneously removing a Cash asset worth $200 from Brown's inventory and placing it in NewCo's. The output function is constructed from the input string through a sequence of steps involving a grammar. As mentioned above, a grammar consists of two types of symbols, "terminal" and "non-terminal". By convention, non-terminal symbols are enclosed by pointy brackets <>, and the parser always begins with a "start" symbol, in this case < `transactions` >.

In addition to these symbols, a grammar consists of a set of overwrite rules that prescribe how to replace any particular non-terminal symbol with other symbols. Our own set of overwrite rules is shown in Fig. 3 under the heading "Grammar". In each line, the left hand side of the "::=" sign is a particular non-terminal symbol, while the right hand side contains a list of the allowed symbol combinations it can be replaced with. The elements of this list are separated by vertical bars '|'.

To illustrate, we begin with the designated "start" symbol < `transactions` > and the first codon value 3. From the figure it is clear that our start symbol can be replaced by two possible combinations of symbols. To decide which overwrite rule to use, it is usual to follow the following protocol. First, divide the codon value c by the number of possible overwrite rules r, and compute the remainder (here $c = 3$ and $r = 2$, so the remainder is 1). Second, add 1 to this value, and choose the corresponding rule in the list. As shown in the figure, applying this formula to overwrite the leftmost non-terminal symbol at each iteration of the process leads eventually to the set of terminal symbols under the heading "Phenotype".

As discussed in Sect. 2.1, all genetic algorithms require a means to rank candidate solutions according to fitness. In the present context, the simplest nontrivial functional form for the fitness function F of a scheme is just $F = G$, where G is the difference between the tax owed before any transactions, and the tax that is assessed afterwards. Under this measure, fitter schemes are those which afford the greatest reduction in tax liability. In the real world, however, it is clear that some schemes are riskier than others, or else cost more to implement, and hence carry a lower expected return. Such considerations will almost certainly affect the likelihood that any particular scheme survives.

It is relatively inexpensive to set up partnerships or other pass-through entities. Since promoters typically charge a percentage of G (Hamersley 2004), this is by far the dominant cost of implementing a scheme. Cost is therefore only a very weak function of scheme structure; as such, it is unlikely to exert much influence on the success or failure of any particular scheme. By contrast, risk of detection (and hence fines) depends rather strongly on the precise manner in which a scheme is implemented, and will therefore tend to determine its long-term viability. These considerations suggest the following refinement of the above fitness measure to account for risk:

$$F = G - E(P) \tag{1}$$

where G is the "tax gap" defined above, P is a penalty assessed upon detection of the scheme, and $E(\cdots)$ denotes expected value. In the real world, for a particular scheme k, $P = P(G) = \phi_k G$, where the coefficient $\phi_k > 1$ is a factor determined by the nature of the scheme (for simplicity, we here ignore fixed fines and limits to penalties that may apply in certain cases). If p_k is the probability that scheme k is detected, then $E(P) = p_k \phi_k G$, and

$$F = G(1 - p_k \phi_k). \tag{2}$$

The $E(P)$ term has the effect of reducing the expected return, and hence fitness, of riskier schemes. Over time, as auditors gradually become more aware of a particular scheme k, p_k increases to the point that the scheme becomes unviable, and promoters are forced to innovate. We model the effect of this term in our genetic algorithm by flagging a set of n suspicious transactions and assigning each of them an 'audit score' a_i such that $\sum_{i=1}^{n} a_i = 1$. The a_i can loosely be considered as capturing the relative probability that a particular flagged transaction will be detected. We then count the number of times f_i that each flagged transaction i occurs within a scheme, and compute the fitness F as $F = G(1 - \sum_{i=1}^{n} a_i f_i)$. This captures the essence of the "real world" situation described above by penalizing instances of suspicious transaction activity in proportion to their frequency and likelihood of detection.

This approach affords us the flexibility to include all kinds of transactions regardless of their strict legality; it is only necessary to assess the degree to which the associated activity is likely to attract the attention of auditors. It also gives us the ability, by manual variation of the audit scores, to determine the kinds of schemes that are likely to proliferate under changes to audit priorities. Such a feature has enormous potential to shape audit policy by helping decision makers explore the implications of proposed changes to existing priorities.

3 Experiments

As an initial step, we applied the genetic algorithm methodology to search for tax evasion schemes involving artificial basis step-up transactions. One such scheme, called *Installment Sale Bogus Optional Basis* or IBOB (GAO 2010), is illustrated in Fig. 4.

This figure shows the simplest possible version of IBOB capturing all the essential elements required to implement the scheme. It depicts a notional tax evader, Mr. Jones, who controls a network of entities consisting of two partnerships, JonesCo and NewCo, and a trust called FamilyTrust. Mr. Jones has become aware that an outside party, Mr. Brown, wishes to buy a Hotel asset owned by a member of his network. The Hotel has accumulated considerable value since its purchase, creating a sizable mismatch between its fair market value and its basis. In order to reduce this mismatch, Mr. Jones manufactures a sequence of trades between some of the entities in his network, trades designed to trigger upward basis adjustments in the Hotel. These adjustments enable Mr. Jones to claim a reduction in the tax owed on his share of the proceeds from the eventual sale of the Hotel to Mr. Brown.

The scheme is structured as follows. Mr. Jones owns a 99 % share of JonesCo, which in turn owns a 99 % share of NewCo. The \$198 value of this latter share derives from the assets owned by NewCo, which for simplicity we assume consists of a single Hotel worth \$200. The basis of the hotel is \$120; if NewCo were to sell this asset straightaway to Mr. Brown, the owners of JonesCo, and Mr. Jones in particular, would be obliged to recognize their share of the \$80 gain. Instead, Mr. Jones arranges for the following sequence of transactions to occur.

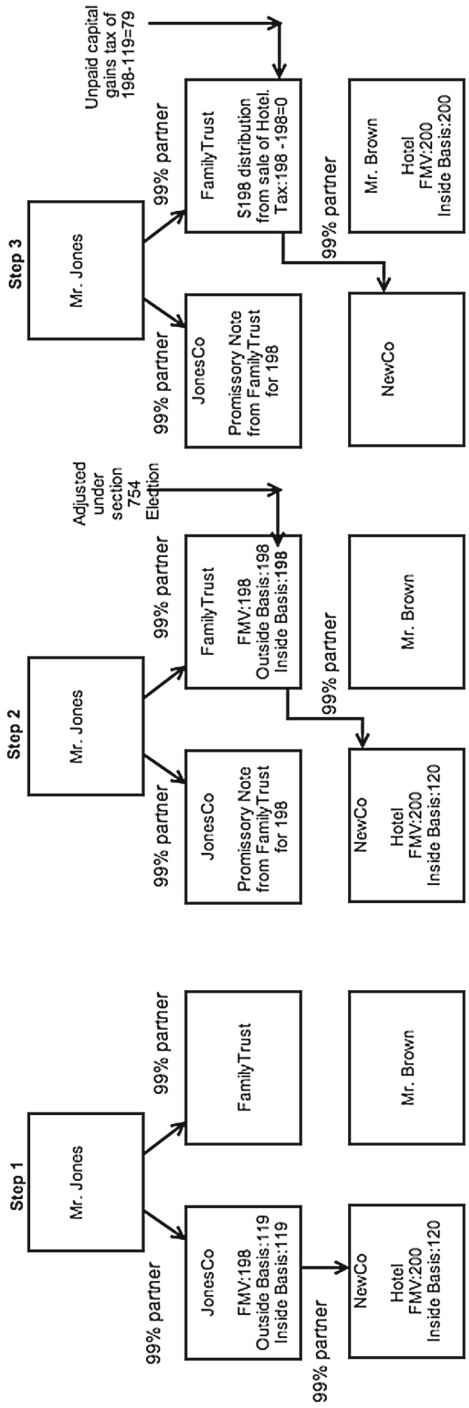

Fig. 4 The steps in the IBOB tax evasion scheme. The basis of an asset is artificially stepped up and tax is avoided by using pass-through entities

First, he causes JonesCo to sell its share of NewCo to an entity called FamilyTrust, of which Mr. Jones is the sole trustee, in exchange for a promissory note with a face value of $198. This event triggers the possibility for a Section 754 Election (CPA 2005) on the part of NewCo. A Section 754 Election is made possible whenever a partnership share is sold by an existing partner to an outside party; since the outside party is essentially paying fair market value for their share of all the assets owned by the partnership, they are not obliged to pay tax on any gains realized with respect to the original basis of those assets. Assuming there is gain, the associated tax will presumably be paid by the selling partner.

To remedy any potential accounting difficulties associated with this fact, the partnership may elect, under Section 754, to adjust the share of inside basis in these assets upward for the purchasing partner. Since Mr. Jones owns a controlling interest in NewCo, he can force the 754 Election and adjust FamilyTrust's share of the Hotel's inside basis upward to $198. FamilyTrust then defaults on its promissory note. Since JonesCo receives no payments, it reports no gain. Further, since JonesCo and FamilyTrust are both controlled by Mr. Jones, JonesCo will never pursue legal action against FamilyTrust. When the hotel is finally sold to Mr. Brown, FamilyTrust receives $198 but Mr. Jones pays no tax, due to the earlier basis adjustment.

The grammar used for the IBOB scheme is as shown in Fig. 3. Note that since `< transactions >` is recursive, the search space is in theory infinite.

3.1 Results

To test our algorithm's ability to construct IBOB schemes, we've conducted two sets of experiments. In the first, we ignored risk and cost by using the fitness function $F = G$, where G is the difference between the tax owed and the tax ultimately paid. In the second, we used the modified fitness function $F = G(1 - \sum_i a_i f_i)$ described in Sect. 2.2 above.

We begin each run by creating the following set of Entities: MrJones, NewCo, JonesCo, FamilyTrust, and MrBrown. These entities are all linked together according to the ownership network depicted in Step 1 of Fig. 4. At inception, each Entity owns a portfolio of Assets including cash, promissory notes, and partnership shares (and, in the case of NewCo, a Hotel). We then proceed through the genetic algorithm steps described in Sect. 2.1 above, which results in multiple transactions between these Entities. If a partnership asset is exchanged, then the ownership network is updated to reflect this; further, each such transaction has the potential to trigger a 754 Election. Promissory notes are treated as annuities in which payments are made in installments; issuers of promissory notes have the option to default, and if they do, no gain is reported and no tax is paid.

For each run, a population of 5,000 schemes is evolved over 10 generations. In the first set of experiments, we found that the algorithm is indeed capable of producing IBOB; one such run is shown in in Fig. 5. The curve lingers near zero until IBOB emerges, at which point the fitness jumps discontinuously to its maximum.

Given the rather limited transaction space available it was surprising to discover that IBOB is only one of a number of optimal schemes with respect to the fitness

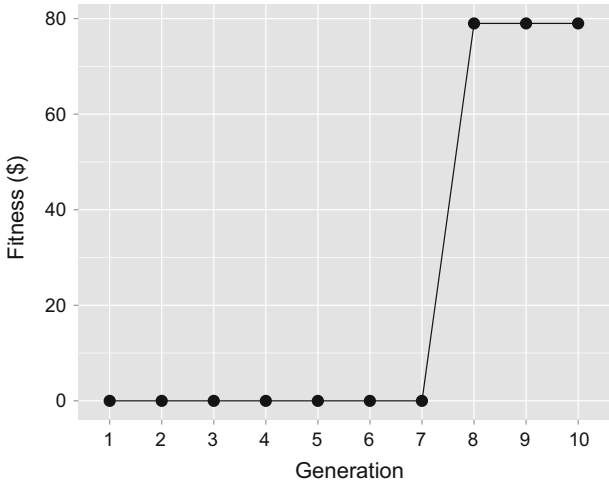

Fig. 5 IBOB GA run. The 'fitness' here corresponds to the tax gap G of the fittest member of the population in a given generation

function G. In fact, the vast majority of schemes yielding the maximum gap within ten generations are not IBOB, but variants like the one described below:

1. Mr Jones buys the Hotel from NewCo with an Annuity and defaults on it
2. Mr Jones sells the Hotel to Mr Brown

In this scheme, no 754 Election is required since Mr. Brown directly purchases the hotel from NewCo. This transaction is clearly absurd; Mr. Jones is essentially buying a Hotel from himself using a promissory note he never intends to repay. Nevertheless, since we do not explicitly proscribe this activity in our code, the algorithm always discovers it in due course. We found many variants like the one above. Examined closely, these variants all make use of one or both of the following transactions: first, an exchange of assets between 'linked' entities, that is, entities connected by one or more degrees of ownership; and second, an exchange of a material asset (like the Hotel) for a promissory note. Since promissory notes are treated as annuities in our code, and since annuities are paid in installments, no tax is collected at the moment the note is issued. If the issuer of the note defaults, no tax is collected at all.

Such variants are clearly very risky propositions since they involve transactions that are more or less openly fraudulent. In principle we could exclude these transactions by adding explicit constraints to the logic of the interpreter module, but such an approach is complex and inflexible. It is far easier simply to penalize these transactions when they do occur rather than to exclude them outright. This has the added benefit of being much closer to the reality of how audits actually work (IRS 2013).

As discussed in Sect. 2.2, we've developed an audit score technique to reduce the fitness of schemes that employ questionable transactions like the ones described above. Our second round of experiments tests the degree to which this technique increases the efficiency of IBOB production by the underlying genetic algorithm. We use the same basic parameters as in the first set of experiments, only now the fitness function is adjusted to penalize schemes that make use of certain 'flagged' transactions. In our

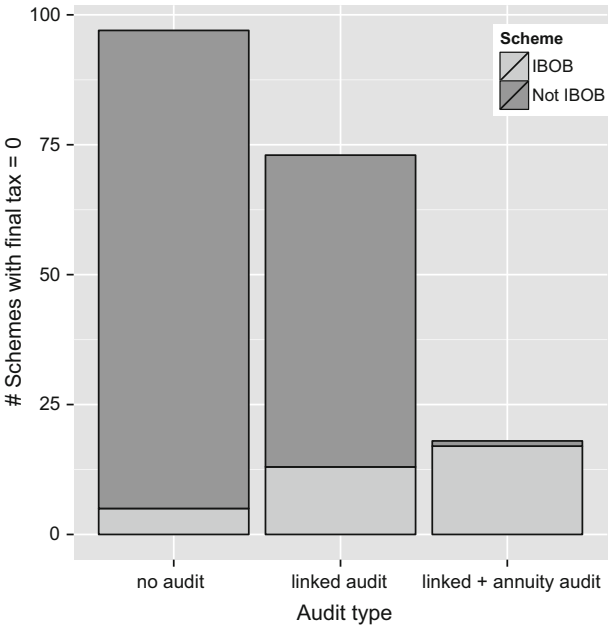

Fig. 6 Second experiment. The graph depicts three sets of 100 runs; of these, the bar contains the number of runs corresponding to schemes where $G = G_{max}$. These are broken down further into IBOB and non-IBOB variants

case, these transactions include: (1) exchanges between entities connected by one or two ownership links, and (2) transactions involving the exchange of a material asset for a promissory note. The results of this experiment are displayed in Fig. 6.

Each bar corresponds to 100 separate runs. In the first bar, no audit scheme is implemented, and the vast majority of schemes that emerge within 10 generations are non-IBOB variants employing either one or both of the risky transactions (1) and (2). In the second bar, all category (1) transactions are penalized, and the proportion of non-IBOB variants surviving ten generations drops, while the proportion of IBOB schemes rises. In fact, the absolute number of IBOB schemes increases substantially. The third bar indicates the effect of penalizing transactions of both type (1) and type (2). Here non-IBOB schemes are practically non-existent, whereas the absolute number of IBOBs increases almost to 20.

4 Conclusion

We have built a functioning end-to-end codebase capable of executing all the steps of the canonical genetic algorithm described above. In particular, we developed a genetic representation with the capacity to generate complex tax evasion schemes like IBOB. This was achieved by separating the representation into two parts: first, a set of Asset and Entity objects responsible for tracking any state changes associated with

transactions, and second, a grammar mapping chromosomes into a set of transaction instructions to be executed by the associated objects.

Our fitness measure employs an audit score methodology in which suspicious transactions are penalized in proportion to their frequency and likelihood of detection. Such a measure incorporates the risk of detection by rewarding those schemes with the smallest expected penalty for a given tax gap G. We demonstrated the ability of this technique to alter the distribution of GA outputs by manually varying the audit scores to penalize specific subsets of transactions. These manipulations led to increasingly efficient production of fitter schemes like IBOB.

We believe our algorithm has the potential to provide valuable insight to auditors seeking to reduce the national tax gap. Enforcement efforts have traditionally been focussed on single financial entities, but unfortunately the most successful schemes involve complex transaction flows within large partnership networks. These schemes are extremely difficult to target without detailed foreknowledge of the associated pattern of activity. Our algorithm can help mitigate this problem by providing detailed exemplars of network-based evasion. Further, it can inform audit policy by enabling decision makers to anticipate the kinds of schemes that are likely to emerge under different enforcement regimes.

Acknowledgments The authors would like to thank Alan Plumley, Kim Bloomquist, Amalia Miller and the journal reviewers for their helpful comments and feedback. This work was supported in whole by The MITRE Corporation Innovation Program. This assistance is greatly appreciated.

References

Allingham MG, Sandmo A (1972) Income tax evasion: a theoretical analysis. J Public Econ 1:323–338

Andrei AL, Comer K, Koehler M (2013) An agent-based model of network effects on tax compliance and evasion. J Econ Psychol 40:119–133

Bloomquist KM (2006) A comparison of agent-based models of income tax evasion. Social Sci Comput Rev 24:411–425

Brabazon A, O'Neill M (2010) Biologically inspired algorithms for financial modeling. Springer, Heidelberg

The CPA Journal (2005) http://www.nysscpa.org/cpajournal/2005/205/essentials/p50.htm. Accessed 28 November 2013

Davis JS, Hecht G, Perkins JD (2003) Social behaviors, enforcement, and tax compliance dynamics. Account Rev 78:39–69

GAO-10-968 (2010) http://www.gao.gov/new.items/d10968. Accessed 28 November 2013

GAO-12-651T (2012) http://www.gao.gov/assets/600/590215. Accessed 28 November 2013

GAO-14-453 (2014) http://www.gao.gov/assets/670/663185. Accessed 26 June 2014

Goldberg DE (1989) Genetic algorithms in search, optimization, and machine learning. Addison-Wesley, Boston

Hamersley M (2004) Interview comments on PBS Frontline documentary "Tax Me if You Can" http://www.pbs.org/wgbh/pages/frontline/shows/tax/schemes/kpmg.html. Accessed 19 June 2014

Hokamp S, Pickhardt M (2010) Income tax evasion in a society of heterogeneous agents—evidence from an agent-based model. Int Econ J 24:541–553

IRS (2006) http://www.irs.gov/uac/IRS-The-Tax-Gap. Accessed 28 November 2013

IRS (2009) http://www.irs.gov/pub/irs-pdf/p3744. Accessed 28 November 2013

IRS (2011) http://www.irs.gov/Businesses/Guidance-for-Examiners-and-Managers-on-the-Codified-Economic-Substance-Doctrine-and-Related-Penalties. Accessed 28 November 2013

IRS (2013) http://www.irs.gov/publications/p556/index.html. Accessed 28 November 2013

Robertson JF, Quinn T, Carr R (2010) Codification of the economic substance doctrine. J Bus Adm Online 9(2):1–7

TaxAnalysts (2005) http://www.taxanalysts.com/www/freefiles.nsf/Files/106TN0453/file/106TN0453. Accessed 28 November 2013

Wright D (2013) Financial Alchemy: how tax shelter promoters use financial products to bedevil the IRS (and How the IRS Helps Them), Valparaiso University Legal Studies Research Paper No. 13–3