Working Paper
ENGLISH ONLY

UNITED NATIONS ECONOMIC COMMISSION FOR EUROPE (UNECE)

CONFERENCE OF EUROPEAN STATISTICIANS

EUROPEAN COMMISSION

STATISTICAL OFFICE OF THE EUROPEAN UNION (EUROSTAT)

Joint UNECE/Eurostat work session on statistical data confidentiality
(Helsinki, Finland, 5 to 7 October 2015)

Topic (ii): New Methodologies for Protecting Data (Disclosure Limitation)

Techniques to apply cell suppression to large sparse linked tables and some results using those techniques on the 2012 (US) Economic Census

Philip Steel, James Fagan, Vitoon Harusadangkul, Paul Massell, Richard Moore Jr., John Slanta, Bei Wang*

* This paper represents the joint work of the Research and Methodology team on cell suppression at the United States Census Bureau[1], Philip Steel is the contact author: philip.m.steel@census.gov.

Abstract: This paper is a follow-up to "Re-development of the Cell Suppression Methodology at the U.S. Census Bureau presented in Ottawa [3]. In this paper, we continue our description of the methodology implemented in our production secondary (complementary) cell suppression software. This includes how we adjusted the model and queueing to handle m primary suppressions (P) in a simultaneous linear programing problem (m-LP). We correct our specification of the model for non-additivity and provide our conventions for handling negative data. We discuss weighting parameters and the mixed success in utilizing them. Finally we describe some of our 2012 Economic Census tabulations and provide a summary of some of the large file (>750,000 P) performance of our new application.

1 Economic Census data and disclosure

Parts of the US Economic Census have been conducted since 1810 [1]. The current five year cycle has been in place since the 1940. The data collection period is roughly

[1] This report is released to inform interested parties of ongoing research and to encourage discussion of work in progress. Any views expressed on technical issues are those of the authors and not necessarily those of the U.S. Census Bureau.

six months. Edit, review and follow-up occur over the following year. Preliminary tabulations occur at the end of collection and the main publications begin approximately two years after the reference period. A wide variety of data is collected with content variation between sectors. The core of the main publication is the Geographic Area Series. Tabulation is attempted for more than 17,000 geographic entities. There are two approaches to structuring the publication. The mining, construction and manufacturing sectors are processed and released for all states at one time. The remaining sectors, including retail and wholesale trade, are published in groups of states on a flow basis.

The disclosure avoidance strategy has been the same since at least the 1939 Census of Manufactures: hide identifiable reports in aggregates [2]. The complexity required to execute that strategy has grown as the publication itself has grown and become interconnected. Cell sensitivity is currently determined by the p% rule. These constitute the primary suppressions; secondary suppressions are required because of table additivity and are selected by a cell suppression program to minimize overall impact.

With the advent of tailored electronic delivery of statistics, in particular geographic profiles, which pull statistics together from a wide variety of sources, Economic Census data has had to reconcile its publication geography with demographic publication geography. This has led to significant over-tabulation of economic data—e.g. a county that supports a variety of demographic statistics (designed for counties) may have very few reportable economic statistics and those may be at a relatively high classification level. However, we make an attempt (and occasionally succeed) to populate the profile of counties and smaller geographic units with Economic Census data. As a result, 60-90% of the nonzero cells in the Geographic Area Series tabulation are primary suppressions.

2 The usage of the LP model

The details of the LP model were covered in [3], so only an outline of the process will be given here. The processing unit for the system is a table group. A table group is a collection of related tables. Cells have two attributes in the model, cost and capacity. Cost is related to cell value and possibly its position in the table. Capacity represents its ability to protect another cell. For example, a previously published cell has capacity zero; its status is '**frozen**'--it cannot be suppressed to protect another cell.

The processing has two phases and two stages within each phase. The first phase we call **base pass**. A queue of "**targets**" is set up. These are primary suppressions (**P**) or complementary suppressions (**C**) carried over from a previous job. When processing a target, we add two equality constraints to the problem; when we are done with that target, its two constraints are removed. In the first phase, stage one, an LP problem is executed and the solver returns a matrix which represent flows or circuits that protect the primary suppression from disclosure via table additivity. This flow may contain unnecessary suppressions. Consequently, a second stage problem is performed to remove the excess. This problem is limited to the solution from the first and uses an inverted cost function. The pattern associated with the target is then committed and

available for subsequent reuse. The next target is processed in the same way. Base pass is done when the queue is exhausted. The accumulated solution is then tested for company protection problems. Company protection problems occur when an aggregate cell (supercell) fails the p-percent rule. This most frequently occurs in aggregates of two cells consisting of one company report from each. Suppressing both cells protects them from outsiders, but each can derive the other's value. A queue of those problems is constructed for the second phase, the **supercell pass**. It is similar to base pass, but also adds an aggregate constraint with each problem and makes some capacity adjustments. In all, the model has four different flavours depending on which part of the process is active.

The procedure we call **skip-P** was essential to our ability to process these files. Skip-P identifies targets that would produce a 0 objective result (already protected). At each commit point, the program identifies and removes already protected cells from the processing queue. I believe Gordon Sande was the first to implemented skip-P; it was re-invented in the course of our research. Section 8 will show the impact of this procedure.

3 An LP cell suppression system for large sparse problems

A number of systems have been developed over the years to do secondary cell suppression. Many are ad hoc and tailored to particular publications. Major systems deal with a variety of tables and organize around the additive relationships which define a particular table. Linked tables are a significant obstacle, but can be represented simultaneously in an LP model [3]. Linked tables where one table is published before another lead to fixed (frozen) publication status. For nearly two decades the U.S. Census Bureau used a program based on a network algorithm (MCF). The network algorithm is limited in the complexity of table sets it is able to handle. Our new LP system has replaced the network based system and is similar to the Statistics Canada LP system. The linear programing (LP) approach handles complexity without a loss in effectiveness (minimal over-suppression and no under-suppression); but it is computationally intensive. Our replacement system is geared for large, linked and sparsely populated tables. Note that sparsely populated tables generally imply density in Ps. It tolerates some non-additivity, negative values and admits frozen cells.

The inputs for our new system are similar to the old system. They consist of relationship files, which describe the tables' additivity, a cell file with the attributes of the individual nonzero cells, and a program parameter file. We converted to a comma-delimited format and extended it to three full dimensions (one relationship file for each table dimension). The program is in C++ and calls the CPLEX solver for the individual LP problems. The major data structure is a graph, where edges indicate a total/summand relationship. This structure both generates the core model consisting of the bounds, the cost and the additive constraints and manages the cell information through the process. The program creates and manages a queue of secondary suppression problems. For each target in the queue, the program modifies the model, the cost and then launches a CPLEX problem. It processes the return, adding new

complimentary suppressions to the accumulating solution, determines the global cost modifications and modifying the processing queue itself.

The ability to coordinate between the data managed by the program and CPLEX's built-in input structure is crucial. Even on moderate size data, the direct interaction in the production program was up to twice as fast as the prototype, which used AMPL (A Modelling Language for Mathematical Programming) as an intermediary. While the problem itself scales faster, larger problems in the prototype also had a secondary scale problem due to the time to translate and transport each problem and solution to and from CPLEX — with additional hazards on memory and the possibility of memory leakage. The footprint of the production program in memory is constant, though dependent on problem size (and may exceed the capacity of a standard PC). CPLEX can use multiple CPU for the barrier method; we did not see usage exceeding 12 CPU and that usage can be limited with an option. CPLEX was our solver of choice, but the code is structured to accommodate a change in solver.

The time it takes a solver to process a problem grows very quickly with the size of the problem. As the size increases (and as the number of 0 cost cells grows) jobs transition from problems most quickly solved by dual simplex to those that require the barrier method. When solution times for individual problems start hitting the five minute mark you begin to see microflows, an initial sign of precision problems. Problems requiring even longer solution (12 hrs) times become swamped with minute flows, algorithmic restarts and ultimately convergence failure. I would note that targets with a protection requirement of one are the first to show convergence problems.

4 Correction of the model for non-additivity

In Steel etal 2013 [3], the model was specified to omit constraints where there was significant discrepancy (greater than p%) between a nominal total and its summands. We reason that any attempt to back into values, using this relationship, would yield an estimate of a top company value with more than p% error. However, the results of the omission were not desirable. If a pattern was able to include cells in that relation (whose constraint is missing in the model), it often did and pattern in tables displaying that relation would appear to be broken. Data would then consistently fail post-processing checks, potentially obscuring actual problems. We also anticipate that this would immediately generate difficult questions from the public (before they have read the fine print in the methodology): why suppress a value that can be backed into? why is the table not additive? We removed that additivity check and allow flow through the non-additive relation as if it were additive.

5 Negative values in data

The premise behind cell suppression is that we can construct a mathematical model of a (positive) table and set up a problem in this model that is able to determine a best suppression pattern. If a table lacks additivity, this model does not quit fit the data. The same sort of discrepancy occurs when the data have negative values—the bounds

in the model are violated. We create, somewhat piecemeal and within the cell suppression set up, an intermediary positive table for which we can set up a proper model. The additive relationships of this table are the same, but the bounds determined by absolute sums and zero. We are constrained by what comes out of the tabulation process i.e., we do not have access to all the absolute sums we would like to have.

We do not scale the amount of protection. One can argue that if there are a sufficient number of respondents, some of whom could be making a negative contribution, that *no* protection is needed, since one can no longer back into company values knowing the cell value. Of course, this assumes that there are enough respondents and that users do not know the degree of dominance by one company of the cell. A useful way to think about this is to consider the p-percent rule one extreme and a simple three company threshold as the other, along some spectrum of suppression rules [4]. We loosen the calculation to allow extra capacity, but do not go any further.

We rank the company contribution to the cell by magnitude, with y1 being the company whose absolute contribution is greatest and y2 to be the sum of ranked collaborators. We define the absolute total of the cell to be the sum of the absolute value of company aggregates with the cell.

$$\text{abstot} = \sum_{i=1}^{n} abs(\sum_{j=1}^{k} y_{i,j})$$

The extra summation is a reminder that the original data is at the establishment level and that some cancellation may occur within company. There has been some debate whether we should protect at the company or the establishment level. For us the question is moot, since we only have access to the company level data.

The remainder, as defined in the p% rule, is computed by

$$\text{rem} = \text{abstot} - \text{abs}(y1) - \text{abs}(y2)$$

where y1 and y2 are the highest ranking company and the sum of the ranked collaborators (by magnitude), respectively. The protection need is calculated then as usual. Note that when you have a mix of positive and negative values that rem tends to be larger and the cell is less likely to be sensitive. See [5] for more on the p% dominance rule.

Due to input constraints, we could not carry this forward into the program; the p% calculation was carried out before cell suppression, and cell suppression does not have access to the full microdata, only the top company and collaborator values. So we define the cell capacity (the bound in the LP problem) to be

$$\text{capacity} = \max(\text{abs}(\text{cell value}), \text{abs}(y1)+\text{abs}(y2))$$

A similar approximation was used when determining sensitivity for company level protection and the capacities for protecting companies in aggregates. Our

implementation departs from the strategy of creating and processing a true absolute table two respects: (1) we may have unaccounted for within company cancellation and (2) we may have underestimate the capacity of the cell. In theory, both approximations could lead to an infeasibility … a protection requirement bigger than the cell's parent total. This did not occur; reasonable p values provide a big margin between totals and protection requirements.

6 Infeasibility

Infeasibility of an LP problem may occur in three ways. If cells have been previously published i.e., frozen with bounds at 0, an infeasibility may result if all possible suppression patterns require some frozen cell or cells. In processing the part of the census that publishes piecemeal by groups of states, with frozen cells being carried over from group to group, we ran into about a dozen infeasibilities. All were resolved before the frozen data were published by identifying and insuring that the cell that was needed for a later suppression was in fact suppressed rather than published. The sequence of state groupings was designed to minimize this sort of problem. The majority of the infeasibilities involved fixed zeros at the lowest publication level. Another infeasibility occurs in m-LP (examined in the next section) when the problem for the selected *m* group does not have a solution. This is easily remedied by automatically reverting to a 1-LP process for that group. An infeasibility of the first type will cause an m-LP failure. That is only discernible after the fact, when one of the 1-LP problems in the reversion is infeasible. The third type occurs in tables with severe additivity problems; in practice, these severe errors have required redress for reasons other than completing the disclosure processing.

7 m-LP

In order to speed up processing, we changed the model to do multiple targets in each optimization. Instead of two target specific constraints, we add two for each of the targets being grouped together. There is no noticeable increase in solve time for a problem in the standard model (1-LP) versus a problem in the m-LP model. If anything, it is usually a little bit faster and gives a little bit better solution. As with the 1-LP, the model forces directionality to the flow variables. The program fixes an alternation (out of 2**m possible alternations) of sign; for some groups of targets this can cause infeasibility or over-suppression. If the targets are well separated (protection patterns do not intersect) then the directionality does not matter and hence infeasibility and over-suppression are avoided. The geographic dimension is huge and we rely on it, with some help, to maintain separation. If an m-LP problem is infeasible we revert to a 1-LP process for that group.

7.1 The m-LP filter

We spent some time trying to find a good, quick way to measure how close two cells are to one another. This is a moderately difficult problem and we did not come up with a solution. Instead, we rely on random draws with a crude filter. Cells were randomly assigned to *m* stacks and a cell was selected from each stack. The question

of whether two cells in a draw are close is a birthday problem, in this case with the bad news that the probability that two members are close is surprisingly high. As each m-group was constructed we did a simple screen, barring a cell if it had a coordinate in common with any previously accepted cell, on any coordinate with sufficient variation. The last condition avoids screening on a dimension that could have as few as three possibilities.

Let's look at an example of how this works in production. If the *m* parameter is 17, the problems seldom achieve that size. The screening logic discards as many as 11 targets from the original draw in the base process – on average giving *m*=9 for successful problems. Since supercells have more of a chance of conflicting, the effective m size for supercells averages out something over 5. Late in each process, the stacks become exhausted and the initial draws are reduced. The example below comes from data that has no inherent infeasibility. In section 9, we also factor in reversion to 1-LP, which for data with a substantial number of infeasibles, has a big effect on the average problem set size.

Figure 7.1 shows the profile of effective m size for a process where it worked well. It shows a good bit of variation, a decline as the queues empty out and a mirror process for the supercell phase. "pno" denotes the order of the solves, for example we have the size of the group on the 1000th solve.

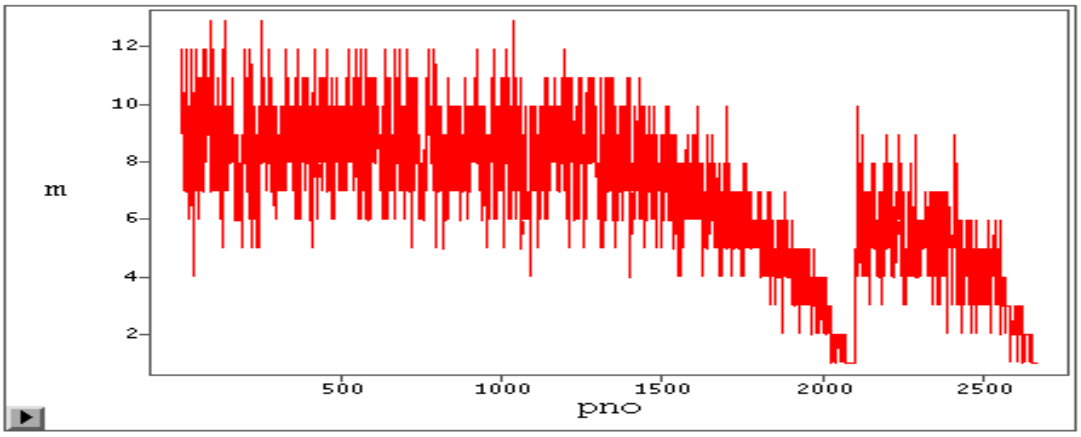

Figure 7.1 Effective m size by solve order.

7.2 The relationship between CPLEX method and m-LP

The default mode for our executable is 1-LP. Small problems are computationally tame and require only dual or primal simplex. We are unable to predict the threshold in problem "size" where barrier becomes advantageous, so we default to the CPU intensive horserace between primal simplex, dual simplex and barrier. Large problems require the use of m-LP and we select *m* to produce a reasonable run time. Our experience has been that users are rather conservative in opting for m-LP, which *may* admit over-suppression, and they will tolerate times on data that transitions midway in the job from predominately dual to pure barrier. Consequently, production runs using m-LP, including those examined in this paper, tend to be on jobs that are 99% barrier. That is, we have data only for m-LP behaviour on "large" problems.

In software testing we observed one of a dozen m-LP run in "m" testing which over-suppressed, but did not observe it subsequently. In the data testing for production, m-LP was consistently better on the number and value suppressed.

7.3 Trial 2 and m-LP

In 1-LP, when trial 1 results in exactly one new C, it's clearly unnecessary to do a trial 2. If the result is two new Cs, we have empirical evidence that, on our data, trial 2 is very rarely effective. These results occur with some frequency and branching the process to omit trial 2 in those circumstances was productive. However, it does not scale properly to m-LP, and further research is required to determine if some analogous rule will work for m-LP. In discussion, we note that a more precise criteria may be applicable there and for 1-LP. Trial 2 is effective when a large observation has shed flow to avoid contributing to the objective function in trial 1. It is a relatively simple to compare the flow through large cost (and capacity) cells to see if the flow matches the protection requirement. If it does, then no "shedding" has occurred.

8 Parameter driven cell suppression

One of the features of the new program is greater flexibility for the user to affect the pattern produced. One was parameter driven, which we called alpha. A general power transformation, x^{alpha} , is applied to cell value prior to the assignment of cost. Alpha set to one gives the standard nominal cost. Alpha in the range of $\frac{1}{3}$ to $\frac{1}{2}$ generally produced better results, often on both the number of cell suppressed and the amount suppressed. However, results are data dependent and we have yet to get a production process to incorporate the testing necessary to set a good alpha. A typical production schedule has already absorbed several hits by the time it gets to the disclosure processing and extra testing is an unattractive prospect. The other feature, flagging particular cells to be more or less important (often by geographic level) was made more accessible than it was under the old system. Since work may be done on this well in advance, that feature saw increased utilization.

9 Some processing results

9.1 Iteration profile over the processing of a single table group

The first cut of payroll in industry GAS had 274,219 Ps and 2812 company protection problems. We used an *m* parameter of 17. The entire process used 5,647 solves. CPLEX provides a stream of data, first giving a time for its pre-solve activity (a proprietary set of procedures to reduce the size of the problem and to select a good starting point for the solve algorithm), the size of the problem to which the input has been reduced, then focusing on the progress of the solve algorithm. For the most part the cell suppression LP problems are uneventful and the progress report consists of periodically providing the iteration count. This gives us a proxy for problem size and difficulty. We capture the last iteration report to estimate the final number of iterations taken by the solve. The iteration profile of the process has strong growth curve as shown in figure 9.1.

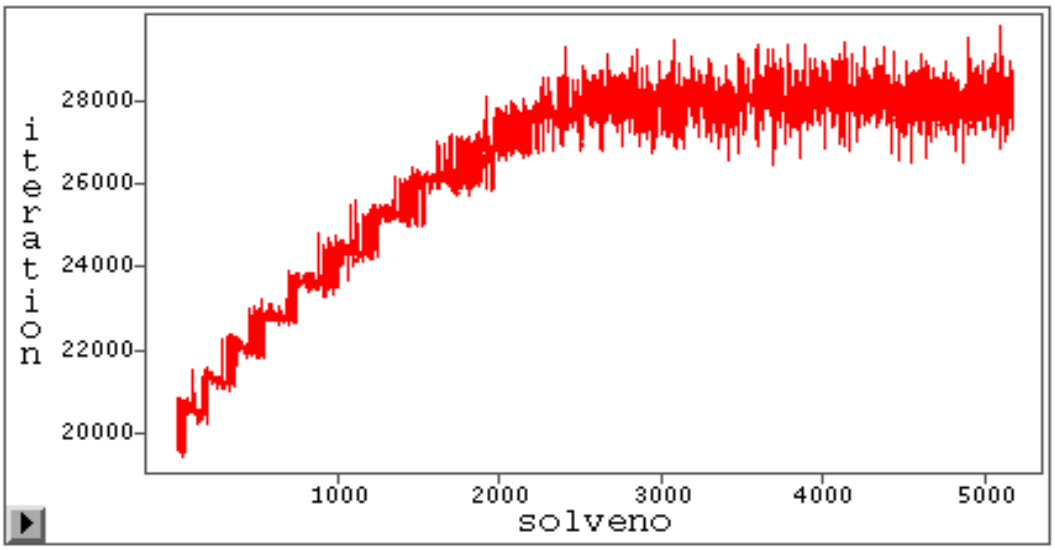

Figure 9.1 Iterations in trial 1 solves plotted against the order of solve, payroll

The company protection phase begins at solve 4156. A single outlier has been removed to improve scale. The problems are nominally the same size (number of constraints before presolve) across the entire process; we attribute the increase in iterations to an increase in the number of 0 cost cells (as Cs are added to the pattern). The step pattern along the fit line is an artefact of the information capture (the count represents the last report of iterations, not the final iteration count). The degree to which it underestimates the final count depends on how long the process continues after the last report. The final count is well approximated by a censored upper envelope imposed on this graph.

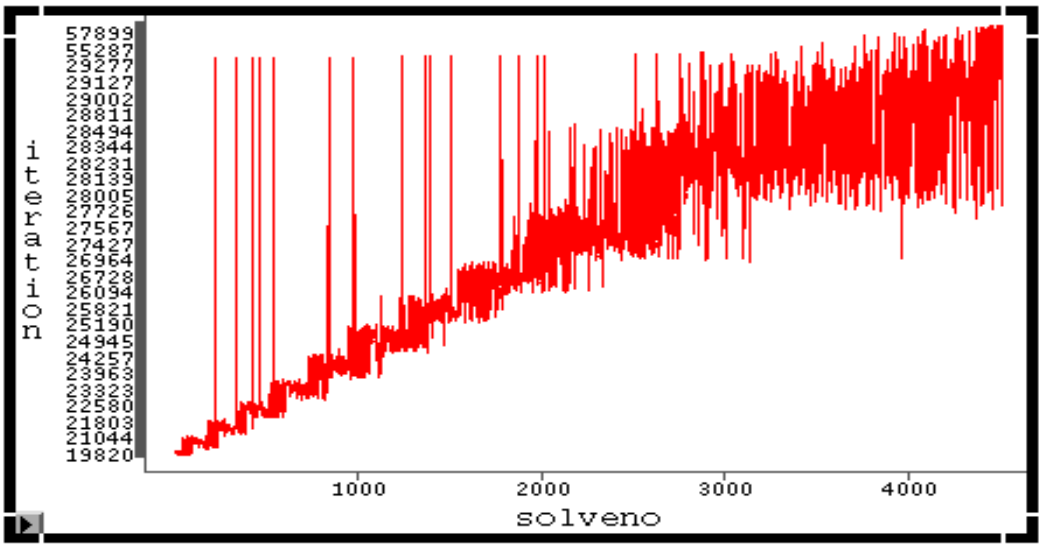

Figure 9.2 Iterations in trial 1 solves plotted against the order of solve, employment.

Every dataset presents new features. The structure of the employment table is identical to the payroll table. One might expect similar behaviour. However, the profile for employment, figure 9.2 has some peculiar spikes. These turned out to be a subset of the problems with a protection requirement of 1. They have long pre-solve times and the result of the pre-solve is a reduction (from the local average) of 25% in row and nonzero counts (CPLEX's internal measure of problem size). The column count is unaffected. The smaller problems are faster in each iteration but wind up taking 50% longer (and have many many more iterations) than neighbouring solves. It is seems likely that CPLEX makes a "mistake" in its pre-solve for this subset.

Other problems have been noted for problems with protection requirement of 1, which we attribute to a flat solution space. For some data, the set of points with protection requirement 1 can be substantial--more than 50%.

9.2 Processing the Geographic Area Series for Industry

The processing of the GAS manufacturing tables was the most difficult part of the Economic Census cell suppression processing. Cells suppression was the dominant process on 36+ CPU for 500 hours. The publication presents results on 15 content variables crossed by geography and industrial classification. Two variables, hours and capital expenditures (cextot), are independent of other variables and were run, with m-LP, as single (2d) tables. The 3d tables could not be run as single jobs. The value-added relationships generate a 3rd dimension with seven variables. The 3rd dimension in employment and payroll each had three variables.

We broke the processing up along the industry classification variable. That variable has a nice tree structure; the alternative, geography, is more complicated. The processing consisted of three rounds. First we processed the cells corresponding to 2- and 3-digit industry classification codes. With results from the first round frozen (fixed publication status), the whole classification tree, without consideration of the two to three digit relation, was run. The program automatically recognizes that the 3-digit trees can each be run separately but will do that in sequence. We manually organized it into four parallel processes, doing the three largest trees and a remainder. Because of the frozen cells, we had some infeasibility. The third round re-ran these infeasible problems in the full (unfrozen) context.

Table 1 summarizes the performance across most of the components of the process, sorted on problem size as measured by time per problem. The performance on the second round remainders has been omitted because each combines a dozen smaller pieces. The value-added third round component was done manually; its problem size exceeded what the system can cope with, both in terms of time per problem and in some instances convergence of individual problems. That component addressed the approximately 500 infeasible targets encountered in round 2 of value-added.

Table 1 gives various cell based measures of the size of the table being processed, the average m-size, how many optimizations were required and the crude average time per problem (runtime divided by optimizations). The first size measure is the simple

Cartesian product of the variables in each dimension. That includes a lot of empty space, so the percentage of cells valued zero is given. Those zero cells are not input to cell suppression. The number of nonzero cells is given and the percentage of these that are primary suppressions. The *m* average combines the effect of the screener and reversion to 1-LP, the number of targets divided by the number of optimizations. The optimizations are m-LP and are one of four types: initial base pattern, final base pattern, initial supercell and final supercell. The four types vary somewhat in size but are not shown or calculated separately.

The reduction from our initial queue of primaries (and the smaller queue of supercells generated after the first pass) to a manageable number of optimizations are a testament to the efficiency of the program, the m-LP processing and the skip-P procedure. Without any one of these, production would not have been possible.

Table	Cartesian Cells	%0	Nonzero	%P	m Average	Optimizations	Time per Opt in Minutes
emp2a	1.9 M	85.4	270,583	89.4	3.7	4465	0.28
emp2c	1.4 M	82.8	244,912	87.3	4.0	4152	0.28
emp1	1.1 M	59.3	462,164	67.9	6.9	5647	0.29
pay1	1.1 M	61.1	432,235	69.6	7.0	5199	0.30
pay2a	1.7 M	85.7	242,104	89.9	3.5	4082	0.30
pay2c	1.4 M	83.0	232,973	87.6	3.6	4278	0.35
emp2b	1.8 M	79.5	374,831	83.3	4.2	7106	0.43
pay2b	1.7 M	79.8	353,361	84.0	4.0	6829	0.47
val2a	3.9 M	87.7	485,699	90.8	5.5	5586	0.96
val2c	3.2 M	84.6	490,203	89.6	5.5	5586	1.91
val1	2.6 M	64.4	922,520	74.9	27.4	2919	2.09
hours	8.7 M	90.0	867,913	71.9	30.2	2688	2.76
cxtot	8.3 M	91.4	711,749	83.2	21.1	2063	4.06
val2b	4.1 M	81.7	747,283	86.2	7.8	7672	4.23
emp3	26.7 M	89.7	2,760,737	--	1	67	105.93
pay3	26.2 M	90.2	2,560,295	--	1	70	206.46

Table 9.1 Table size, Nonzeros, optimizations and time per optimization.

The numbers come from internal counts generated by the program after unduplication. The first line reads: "The second round of processing on employment variables on the second largest of the 3 digit industries was for a table that has almost 2 million cells when represented as a Cartesian product; 85% of those entries are zero. The cells passed to the secondary suppression program numbered 270,000, almost 90% of which where primary suppressions. Averaged across the whole process we did somewhat under four secondary/company protection suppression problems at a time. The whole process, initial pattern, pattern reduction, initial supercell pattern and super cell reduction was accomplished in roughly 4500 optimizations, where each optimization (averaged, with overhead) took about 15 seconds." The most remarkable feature is the action of skip-P: the queue of targets traversed in the job on the first line (emp2a) was around 250,000 cells or aggregates most of which would generated two optimizations in 1-LP. That is, in combination with m-LP, it reduced the computational burden by a factor of 100.

10 Conclusion

This paper describes the problems encountered in the processing of our largest, most challenging, Economic Census tables. Large scale processing is possible with the LP methodology, although these largest problems are at the frontier of what is possible under our methodology with an off-the-shelf solver and standard hardware. Those problems were overcome with patience and the involvement of staff from different areas. In our post-processing analysis, a number of inefficiencies have been identified which could reduce the processing time by as much as 25%. The effort has also generated some ideas on how to avoid the processing crunch altogether; we look forward to researching them over the next couple of years.

The old system had reached-or possibly exceeded-the limitations of its basic methodology. The new system is more readily integrated into the modern computing environment and the new methodology can be extended to handle more complexity and dimensionality.

The adoption of the new system at the U.S. Census Bureau has largely been a success in production, particularly with the smaller survey data. The flexibility of the new system, its better integration into the publication process and the increase in institutional knowledge of how to use it are the measures of its success. Its development will continue as more surveys and new tables enter the publication cycle for the first time.

References

[1] https://www.census.gov/history/www/programs/economic/economic_census.html

[2] Manufactures 1939, Volume III, Reports for States and Outlying Areas, United States Department of Commerce, p7

[3] Steel, P. et al *Re-development of the Cell Suppression Methodology at the US Census Bureau*. Joint ECE/Eurostat Work Session on Statistical Data Confidentiality October 2013

[4] Giessing, S. Protection of tables with negative values. CASC report on the ESSNet SDC website, (ca 2010).

[5]Federal Committee on Statistical Methodology (1994) Statistical Policy Working Paper 22 (Revised 2005)-Report on Statistical Disclosure Limitation Methodology NTIS PB94-165305.

[6] B. Wang Improve LP Process in Cell Suppression, Proceedings of the Government Statistics Section, American Statistical Association, Alexandria, VA (2013)