WP. 44
ENGLISH ONLY

**UNITED NATIONS STATISTICAL COMMISSION and ECONOMIC COMMISSION FOR EUROPE**
**CONFERENCE OF EUROPEAN STATISTICIANS**

**EUROPEAN COMMISSION**
**STATISTICAL OFFICE OF THE EUROPEAN COMMUNITIES (EUROSTAT)**

**Joint UNECE/Eurostat work session on statistical data confidentiality**
(Geneva, Switzerland, 9-11 November 2005)

Topic (vi): Software for statistical disclosure control

# SUDA: A PROGRAM FOR DETECTING SPECIAL UNIQUES

**Invited Paper**

Submitted by the University of Manchester, United Kingdom[1]

[1] Prepared by Mark J Elliot, Anna Manning, Ken Mayes, John Gurd and Michael Bane.

# SUDA: A program for Detecting Special Uniques[1]

Mark J Elliot[**] Anna Manning[*] Ken Mayes [*] John Gurd[*] and Michael Bane[***]

[*] School of Computer Science, University of Manchester, Manchester M13 9PL UK.
[**] Centre for Census and Survey Research, University of Manchester, Manchester M13 9PL UK.
[***] School of Earth, Atmospheric and Environmental Sciences, University of Manchester, Manchester M13 9PL UK.

**Abstract:** The importance of being able to classify records according to disclosure risk is well understood; Skinner and Holmes (1998), Fienberg and Makov (1998). One concept for so classifying records is called special uniqueness; see Elliot (2000), Elliot et al (2002), Manning and Haglin (2005). This paper describes SUDA (Special Uniques Detection Algorithm) which is both a set of computer science algorithms and indeed a fully functioning software system for detecting and grading special uniques. Section 1 describes the basic design principles behind the sequential SUDA algorithm. Section 2 describes the software (now in use at the UK Office for National Statistics and Australian Bureau of Statistics). Section 3 describes recent advances (i) in parallelising SUDA and improving the algorithm so that cross-classifications of up to 60 variables can be comprehensively analysed (ii) in developing a version of SUDA for Grid computing.

## 1 Introduction

The principle of being able to classify microdata records according to their disclosure risk is now axiomatic within the SDC field Skinner and Holmes (1998), Fienberg and Makov (1998). Within this paper we describe a software system entitled "SUDA" that provides such record detailed assessment broken down by record, variable, variable value and by interactions of those.

The basic principles behind the SUDA system are described in section 2 and the current version of the window implementation of this software (available as freeware under restricted license) is described in section 3.

Through collaboration of SDC researchers and computer scientists in Manchester efficient search algorithms have been produced which enable special uniques analyses of very large keys at unlimited variable interaction levels, in real time. The use of grid computing to further improve the efficiency is also being investigated. These new methods are described in section 4.

[1] The research described and software development in this paper has been supported by the ESRC, EPSRC and the UK Office for National Statistics.

## 2 The Special Uniques Methodology

The concept of the "special unique" was coined by Elliot et al (1998). The behind the concept principle is that a microdata record which is sample unique on courser, less detailed information is more risky than one which is unique on a finer, more detailed information. A particular case of that is where a record which is sample unique on a set of variables K and is also unique on a subset of K. Such a record is called a *special unique*, with respect of variable set K.

Extensive empirical work (Elliot 2000, Elliot and Manning 2001, Merrett et al 2005) has shown that special uniques are more likely to be population unique than random uniques. Further work (e.g. Elliot et al 2002) has shown that it is possible to classify special uniques according to the size and number of subsets which are unique minimal sample uniques (MSU) and that such classifications are correlated with the reciprocal population equivalence class, which is a generally accepted measure of underlying risk.

### 2.1 The basic SUDA method.

| Notation | Description |
|---|---|
| **ATT** | Total number of attributes in the dataset |
| **REC** | Total number of records in the dataset |
| **M** | User-specified maximum size of attribute set |
| ***(n-1)*-subset** | Subset of size *n-1* |
| **R** | Position of record in the dataset, where $1 \leq R \leq REC$ |

**Table 2.1: Notation**

SUDA is designed around the observation that 'Every superset of a unique attribute set (minimal or otherwise) is itself unique' (referred to as the *Superset Relationship;* Elliot et al. 2002). SUDA will be described in the following sections using the Superset Relationship as a basis for classifying the risk associated with each measure.

SUDA incorporates the Superset Relationship into the attribute set generation process in order to reduce the amount of record comparisons that are necessary. All attribute sets with the same prefix of size P[2], where $1 \leq P \leq M$, are generated in succession so that any superset of a unique prefix at a given record can be ignored immediately without the need to revert to stored information. Given ATT=6 and M=4, with attributes labelled A, B, C, D, E, F, the beginning of the attribute set generation process for SUDA would be as follows:

[2] In general, for an attribute set A containing attributes $a_1,...a_r$ (with $1 \leq r \leq M$), a prefix of size P of A where $1 \leq P \leq r$ contains the first P attributes $(a_1,...,a_p)$ of A.

A, AB, ABC, ABCD, ABCE, ABCF, ABD, ABDE, ABDF, ABE, ABEF, ABF, AC, ACD etc ...

An example of the incorporation of the Superset Relation can be given as follows. Consider attribute sets with prefix ABC: that is ABC, ABCD, ABCE, ABCF from the above listing. If attribute set ABC is found to be unique at record R all supersets with ABC as their prefix can be ignored for R as they are not minimally unique. As all such supersets appear directly after ABC in the above sequence this process can be carried out immediately without the need to check stored information and has the effect of reducing the number of records that need to be considered for each attribute set while at the same time minimising memory usage.

## 2.2 Record grouping procedures

A grouping method is used in SUDA to collect together records with identical values for a given attribute set. This localises the records required for any given search and minimises the amount of memory usage that is necessary to identify minimal uniques. Figure 2.1 shows a dataset with 20 records and three attributes (A, B, C) and illustrates the check for potential uniques of attribute sets A, AB and ABC.

The records are initially grouped in terms of attribute A, by placing values of R in a 2 dimensional matrix according to their value of A, as shown in the bottom left of Figure 2.1. The records are then rearranged according to the results of this process into partitions, as shown in the second dataset configuration in Figure 2.1. Any partition with one member represents a minimal unique for A and this record can be removed from the grouping process; as A is a single attribute no check is required for the uniqueness of its subsets.

Attribute set AB is then checked by considering attribute values for B in each partition of the second dataset configuration in Figure 2.1.

If the dataset had more than three attributes (i.e. ATT≥M>3) the grouping procedure described above would be applied recursively to each of the partitions in the second dataset configuration in terms of attribute B and the resulting partitions placed in the third dataset configuration. Any partition containing just one record number would represent a potential unique and this information would be saved (in order to check for minimal uniques later) and the record number would not be placed in the third dataset configuration (as all supersets of this attribute set for this record would be unique).

However, ATT=M=3 and Figure 2.1 demonstrates how the uniqueness of attribute sets of size M-1 and M can be found more quickly.

For attribute sets of size M-1 a one-dimensional matrix *ONE* is used to identify potential uniques. For the $(M-1)^{th}$ attribute value of each record (in this case, attribute B) the corresponding record number (R) is placed in ONE according the value of B: e.g. a record with B=1 has its number (R) placed in the first cell of ONE and in the second cell if B=2. If more than one record is placed in any one cell its value cannot be unique and a value such as '-99' (or 'x' in Figure 2.1) represents this information. When all records in a partition have been checked all cells of ONE are scanned and any that don't contain 'x' or '0' contain values of R for records that are potentially unique for AB. The contents of the partition are then copied to the third dataset configuration, leaving out any records that were found to be potentially unique for AB. This procedure is repeated for all partitions in the second dataset configuration.

Attribute set ABC is then addressed. Records in dataset configuration 3 are considered on a partition level basis as before. For attribute sets of size M, record numbers are placed in a two-dimensional matrix *TWO* in which each cell corresponds to the values for the $(M-1)^{st}$ and $M^{th}$ attributes (B and C) of each record. For example, if B=1 and C=2 for a record its number (R) is placed in the cell at the intersection of the first row and the second column of matrix TWO. As with ONE, when all records of a partition have been considered all cells of TWO are checked and if any do not contain 'x' or '0' then these record numbers represent records that are potentially unique for attribute set ABC. All records from this partition are then copied to the fourth dataset configuration in Figure 2.1 omitting the potential uniques (which are stored as before). This procedure is repeated for each partition of the third dataset configuration.

Figure 2.1 has been designed to explain the grouping procedures used by SUDA but suggests that the entire dataset is re-grouped for each attribute set. However, SUDA does not physically re-group the records of the dataset at any stage but uses a matrix (referred to as *Group Matrix*) to store values of R within each partition [Elliot et al. 2002].

### 2.3 The check for minimal uniqueness

All potential uniques of size $n \geq 2$ must be checked for the non-uniqueness of all their *(n-1)*-subsets to ensure that a minimal unique has been found. This has the potential to lead to very high memory requirements. However, due to the generation of attribute sets according to their prefixes the information can be retrieved from the Group Matrix and involves the use of a hash table Elliot et al. 2002.

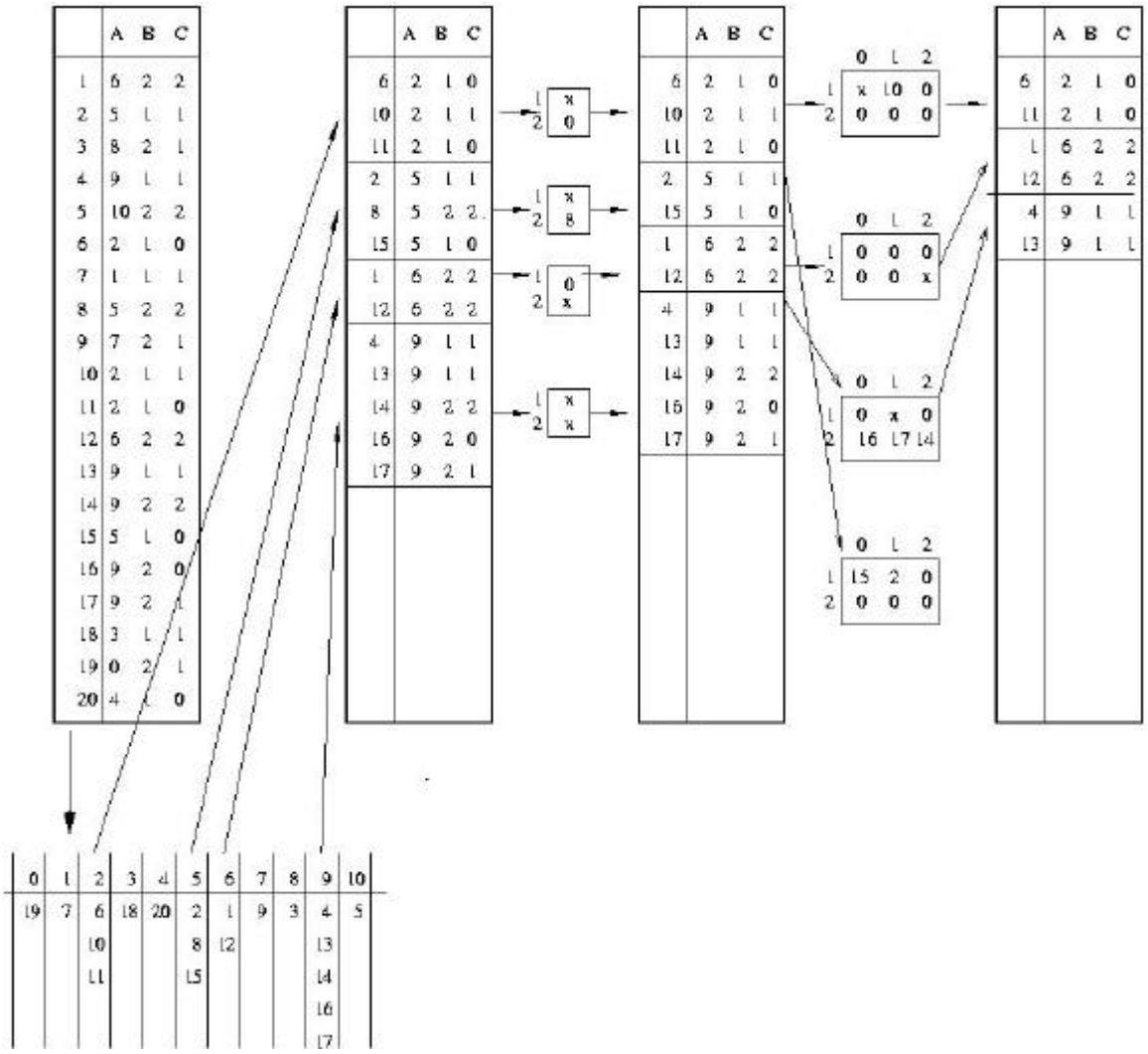

**Figure 2.1**: Record grouping process for SUDA

## 2.4 Combining information from the lattice

### 2.4.1 Generating the intermediate SUDA metric

Once all minimal uniques have been found the following characteristics are important in the detection and grading of special uniques:

**The size of minimal uniques:** The smaller the size of the MSU within a record the more 'risky' the record is likely to be.

**The number of minimal uniques per record:** The larger the number of MSUs contained within a record the more likely the record is to be 'risky'.

These observations are used to code records according to their potential 'risk' as follows:

Let Ξ = {x1,…, xn} be a set of distinct literals, or attributes. The space of possible sets X? ? ?an be visualised as a lattice - Figure 2.2 shows the case when Ξ = {1, 2, 3, 4, 5} and ATT=5. (This approach is used to describe the search space for association rule discovery [Zaki et al. 1997]).

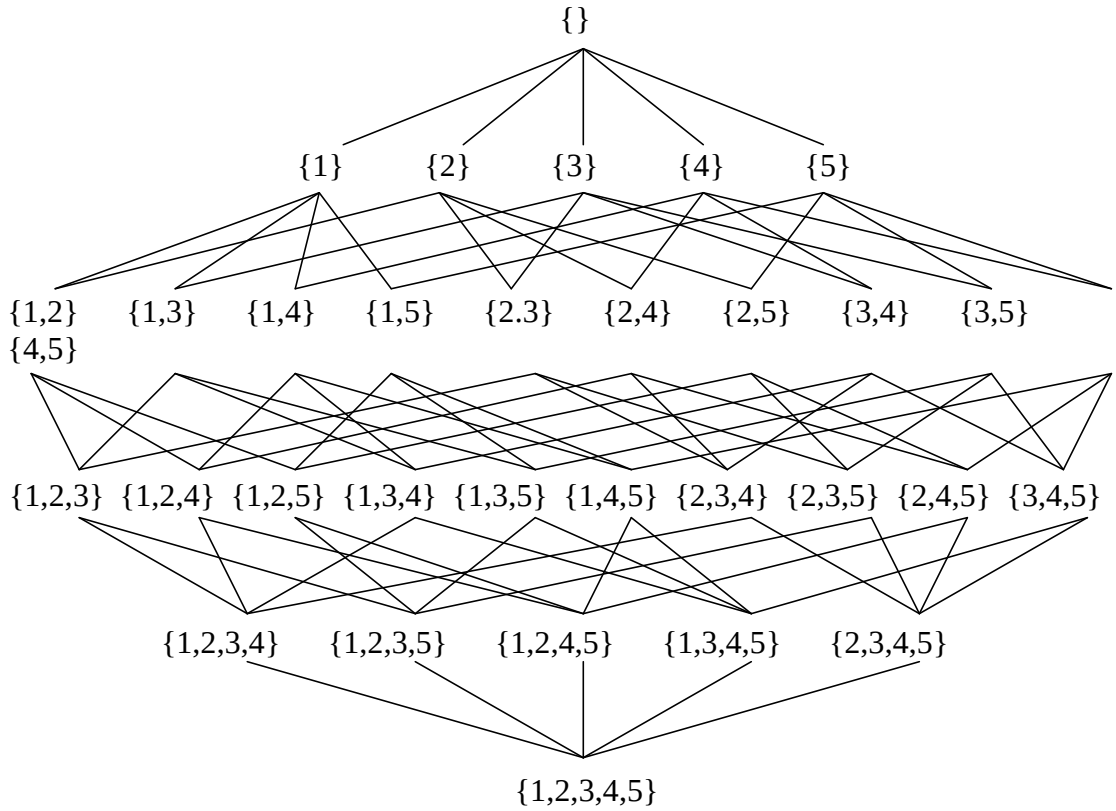


**Figure 2.2 Lattice for** X **= {1, 2, 3, 4, 5}**

Such a lattice can be used to describe the space of all possible subsets of a record.

A judgment needs to be made about the relative risk of MSUs of different sizes. For example, a large number of MSUs of size 3 may be regarded as more risky than one MSU of size 2. The above lattice structure is used to allocate a weight for each record so that the MSUs can be compared.

For each MSU X of size |X|=k contained in a given record R, where $1 \leq k \leq ATT-1$, the Intermediate SUDA metric (IS metric) can be computed by counting the number of distinct 'paths' from X to the bottom of the lattice.[3] This can be represented as:

$$\#\,paths = \prod_{i=k}^{ATT-1}(ATT - i) = (ATT - k)!$$

If k=ATT the number of distinct paths is zero (i.e. no supersets). To avoid giving zero scores to records containing MSUs of size ATT a value of '1' is applied.

In SUDA, the MSUs often have a user-specified maximum size (M). Figure 2.3 shows an adjustment to the lattice in Figure 2.2 when M=2. Here, all the distinct paths from MSUs of size 2 are considered - this has the effect of cutting Figure 1 below the sets of size 2 and only including paths through the lattice from this point.

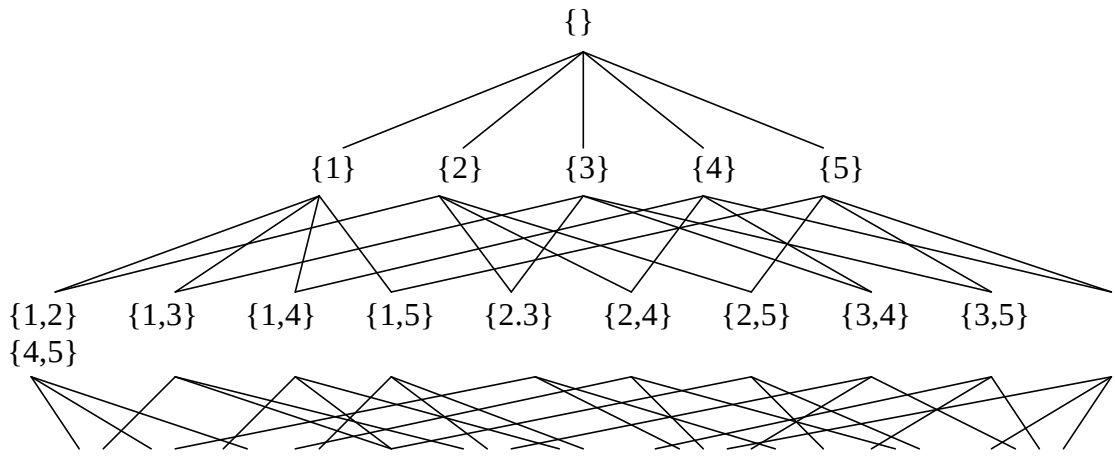


**Figure 1.3** Sub-lattice for M=2

In this case, the number of distinct 'paths' below a given set X of size k where $1 \leq k \leq M$ can be represented as:

[3] Clearly this is just one way in which the MSU information could be combined. It is computationally principled.

$$\#paths = \prod_{i=k}^{M} (ATT - i)$$

The above treats each record-level MSU independently – the scores for each record-level MSU are added together to give the final score for the record.

### 2.4.2 Using the combined information

There are many ways that the IS metric can be used One is to generate a proportion of lattice measure from the number of possible paths through the lattice structure given by ATT! The proportion of lattice statistic represents the IS metric as a proportion of this total:

Proportion of lattice at record R = (IS metric at R / ATT!)

An alternative is to use the data intrusion simulation output metric (see Skinner and Elliot 2002) to generate the total number of population units corresponding to the sample uniques and then to distribute them in some manner dependent upon the IS metric. This method known as DIS-SUDA produces estimates of intruder confidence in a match against a given record being correct. This is closely related to the probability that the match is correct given assumption of zero data divergence. See Elliot (2002), for a further discussion of the interpretation of this metric. The advantage of this method is that it relates to a practical model of data intrusion, and it is possible to compare different values directly. The disadvantages are that it is sensitive to the level of the max MSU parameter and is calculated in a heuristic manner. However, the method has been extensively tested and produces very good results when the max MSU size is large (Merrett et al 2004) and the number of key variables moderate. The proportion of lattice measure is more robust when the max MSU size is lower (for example when conducting a comprehensive rather than scenario based analysis.)

### 2.4.3 Description of risk at record-level and database-level

**Record Level**
In many records there are a small number of attributes that occur in a large proportion of the MSUs as illustrated by the following example.

**Example:** Table 2.1 shows twelve attribute values for an imaginary record. Table 2.2 shows the corresponding MSUs for these attributes.

| Attribute number | Attribute name | Attribute value |
|---|---|---|
| 1 | Age | 80 |
| 2 | Economic Status | Employed full-time |
| 3 | Ethnic Group | White |
| 4 | Limiting Illness | No |
| 5 | Marital Status | Divorced |
| 6 | Sex | Male |
| 8 | Cars | 2 |
| 9 | Tenure | Owner occupier |
| 10 | No. of residents | 4 |
| 11 | No. of Children | 0 |
| 12 | No. of Pensioners | 1 |

**Table 2.1**: Attribute values for imaginary record

| Size 2 | Size 3 | Size 4 | Size 5 |
|---|---|---|---|
| 1 2 | 1 6 9 | | 2 5 6 8 11 |
| 1 5 | 5 8 12 | | |
| 1 8 | | | |

**Table 2.2:** MSUs for imaginary record in table 2.1

| Variable | Occurrence of MSUs of size: | | | % of MSUs affected of size: | | |
|---|---|---|---|---|---|---|
| | 2 | 3 | 5 | 2 | 3 | 5 |
| **1** | 3 | 1 | 0 | 100.00 | 50.00 | 0 |
| **8** | 1 | 1 | 1 | 33.33 | 50.00 | 100.00 |
| **5** | 1 | 1 | 0 | 33.33 | 50.00 | 100.00 |
| **2** | 1 | 0 | 1 | 33.33 | 0 | 100.00 |
| **6** | 0 | 1 | 1 | 0 | 50.00 | 100.00 |
| **9** | 0 | 1 | 0 | 0 | 50.00 | 0 |
| **11** | 0 | 0 | 1 | 0 | 0 | 100.00 |
| **3** | 0 | 0 | 0 | 0 | 0 | 0 |
| **4** | 0 | 0 | 0 | 0 | 0 | 0 |
| **7** | 0 | 0 | 0 | 0 | 0 | 0 |
| **10** | 0 | 0 | 0 | 0 | 0 | 0 |

**Table A.3:** Relative impact of attributes for record 1080324

Attribute 1 is the most prevalent, occurring in all 3 MSUs of size 2 and one of the two MSUs of size 3. The percentage contribution to each attribute to MSUs of each size is shown in Table 2.3.

Let $S_i$ be the IS metric for MSUs of size i, as shown in 2.5.

The total IS metric, $S_R$, for record 1080324 is given by:

$$S_R = 3 \times S_2 + 2 \times S_3 + S_5$$

The contribution to the IS metric for each of the attribute values in record 1080324 are calculated by using the contribution percentages in Table 2.3. For example, the contribution to the IS metric of attribute 1 at record 1080324 ($S_{R1}$) is given by:

$$S_{R1} = 1.0 \times 3 \times S_2 + 0.5 \times 2 \times S_3$$

**Database Level**

The percentage contribution of each attribute value to the IS metric at database level is found by:

1. summing the contributions of this attribute value at record level over the whole file (call this value $T_V$) – for example, the contribution to the IS metric at record-level of every occurrence of AGE=24
2. summing the IS metric ($S_R$) for each record over the whole file (call this value $T_S$)
3. finding $T_V$ as a percentage of $T_S$.

The percentage contribution of each attribute to the IS metric at database level is found by:

1. summing the contributions of this attribute at record level over the whole file (call this value $T_A$) – for example, the contribution to the IS metric at record-level of every occurrence of AGE.
2. summing the IS metric ($S_R$) for each record over the whole file (call this value $T_S$).
3. finding $T_A$ as a percentage of $T_S$.

# 3 The Software

The SUDA algorithms described above have been implemented as a windows application. The application has a simple two window interface; input and output.

The input window, allows the user to specify the dataset, key variables, and the parameters for the run, such as the Maximum MSU size (smaller is faster but less accurate), sampling fraction of the dataset and so on. The input window is shown in figure 3.1. Output is sent to the output window and also to user specified file.

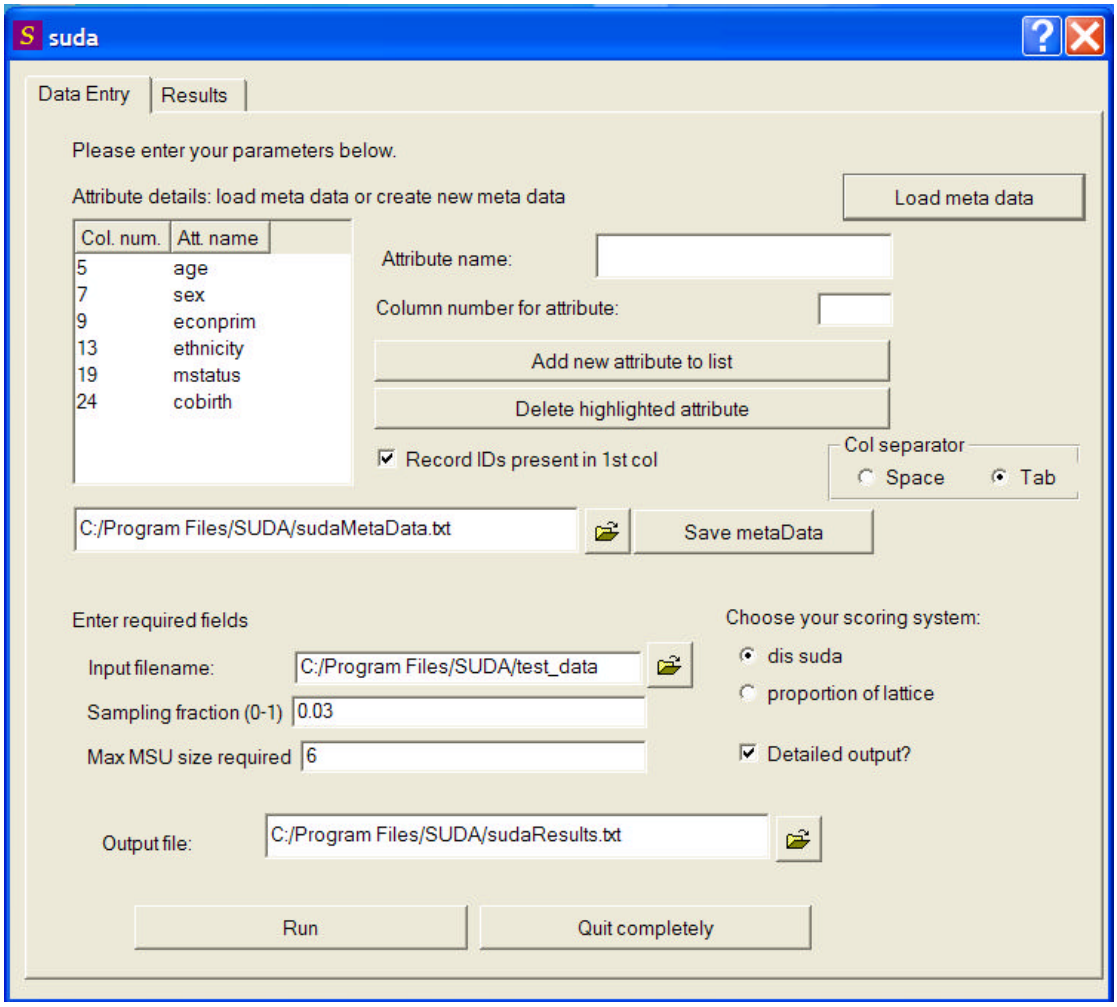


**Fig 3.1** The SUDA input interface.

### 3.1 Description of Output

The output is divided into three parts. The first part contains summary information on the run, the most useful part of the output is the DIS score which provides a file level measure of the disclosure risk.

The second part of the output is the record by record output. The columns of the record level output are:

**1) ID:** if the user has ticked the "ID in first column" tickbox then it this ID, which appears here. If the user has not then SUDA will have assigned integer numbers to each record in sequence one per row, effectively providing a pseudo id.

**2) IS metric:** This is total IS metric calculated as described in section 2.

**3) Scoring metric:** The 3rd column contains either the Proportion of lattice metric or the DIS-IS metric depending on which the user asked for.

**4->N) MSUs:** The sequence of columns after the output metrics give the number of MSUs for the record of each size up to the number the user specified.

**N+1 -> N+K) Contribution percentage:** The final set of columns are headed with the variable name with each of the variables the user has chosen. These columns record the percentage contribution of each variable to the total IS metric. This is simply the IS metric for the MSUs involving that variable over the IS metric for the record.

The third part of the SUDA output is the cross-file breakdown of the IS metric by variable and value. This allows the user to assess where the risk is concentrated within the file. Below is an example for attribute/variable contribution. In this example the available age is contributing. Below is an example for the attribute value contribution for variable age. For both types of output the contribution is the percentage of the total IS metric across the whole file which arises from MSUs involving the attribute or attribute value.

**Example Attribute contribution**

| | | |
|---|---|---|
| col#2 att 'age' | percentage contribution | 88.8954 |
| col#3 att 'sex' | percentage contribution | 14.5084 |
| col#4 att 'mstat' | percentage contribution | 26.7168 |
| col#5 att 'econpr' | percentage contribution | 43.2581 |
| col#6 att 'residents' | percentage contribution | 47.5376 |
| col#7 att 'depchild' | percentage contribution | 26.2359 |

**Example Attribute value contribution output**

| | | |
|---|---|---|
| col#2 att 'age'=0 | percentage contribution | 0.2813 |
| col#2 att 'age'=1 | percentage contribution | 0.4001 |
| col#2 att 'age'=2 | percentage contribution | 0.5090 |
| col#2 att 'age'=3 | percentage contribution | 0.3256 |

etc,…..

# 4 Current and Future Work

## 4.1 The SUDA 2 Algorithms

The SUDA system has greatly increased the depth of risk assessment possible; this was demonstrated by its application to data releases from the 2001 British Census. However, due to the demanding levels of execution time required to find all MSUs in stage one of SUDA, this algorithm is restricted to small datasets, particularly in terms of the number of columns that they possess. This problem formed the motivation for the development of a new algorithm, SUDA2.

SUDA2 improves SUDA using several methods. Firstly a new approach is used to provide a more dynamic representation of the search space for MSUs. Secondly, further properties of MSUs are identified and are used to design improved pruning strategies. Thirdly, a more efficient traversal of the search space is employed.

SUDA2 has the ability to identify the boundaries of the search space for MSUs with an execution time which is several orders of magnitude faster than that of SUDA. Not only will these developments provide statistical agencies with a much faster tool to work with, but the ability to assess microdata with many more variables than before will now be possible.

## 4.2 Grid Hiperstad[4]

The efficiency of the SUDA2 algorithm means that it becomes feasible for large amounts of data to be searched for large patterns. However, as searches increase in size, it is likely that execution time on a single machine will ultimately become prohibitive. Thus it is sensible to provide an infrastructure that allows such applications to execute in a distributed fashion over a heterogeneous network of computers. The aim of the *GridHiPerStaD project* is to produce a prototype software framework for running statistical disclosure applications on a Grid of computers. It is based on the approach of the PerCo performance control system (Mayes et al., 2005).

[4] GRID based HIgh PERformance computing STAtistical Discolure risk analysis

The nature of the suda2 algorithm allows the entire search to be split into subsearches, each of which can execute on a separate machine (though in the present incarnation of the algorithm the data must be replicated). In general, work on what might be termed ``divisible work'' applications fall into two paradigms: master-worker (e.g. Condor MW; Goux et al 1995) and divide-and-conquer (e.g. Blumofe et al (1995). On the whole, the existing systems seek to be paradigm-specific rather than application-specific. That is, they represent efforts to allow application developers to fit a suitable application into the provided paradigm framework.

There is a set of potential problems when considering an application such as Suda2 for distributed execution on a Grid. The available machines are of diverse architectures and capabilities, and may have varying load. On the other hand, the Suda2 subsearches are of unpredictable duration, being related to the nature of the search subspace data rather than its size. Thus in order to optimise application performance, the GridHiPerStaD framework must be adaptive. For example, it must cope with the situation where an unexpectedly large subsearch is executing slowly on a computer that has a heavy multiuser load.

There is some evidence in the literature of a trend to recognise complexities introduced by heterogeneous and unpredictable platforms and applications. For example, in the master-worker system of Kee and Ha (1998) the master is able to redistribute allocated work at runtime. In the work-stealing paradigm there is, for example, the topology-aware random stealing of the SALSA actor-based system, which migrates actors according to communication overhead; Desell et al (2004).

The GridHiPerStaD framework is attempting to make available both master-worker (i.e. centralised scheduling) and work-stealing (i.e. distributed scheduling) {\em mechanisms}. The *policies*, which determine how these mechanisms are used, will be application-specific. That is, the system should be capable of producing any hybrid between master-worker and divide-and-conquer in order to optimise application performance.

Additionally there are facilities for recovering from sub-optimal deployment of work. In the case where the search of a subspace is taking too long, the GridHiPerStaD system can cause the sub-search to be "checkpointed", and the remaining search migrated for resumption on a faster machine. It should also be possible to divide, at runtime, computationally demanding subspace searches.

Such a flexible approach may be necessary where both the computational demands of the application and the computational capabilities of distributed resources may be unpredictable. That is, in such a dynamic scenario, a single scheduling algorithm or paradigm may not be consistently optimal. The performance-orientated scheduling

policy of the system may have to adapt, and this must be underpinned by a number of mechanisms.

## Summary

The SUDA system provides increasingly sophisticated methods for disclosure risk assessment of microdata. The method has now been implemented as a windows software package which is in use in three national statistical agencies.

The method is in a continual sate of refinement and enhancement, both in terms of its Computer Science and the SDC algorithms on which it is based. New sophisticated versions are close to completion. With the possibility of GRID enabled versions in the offing it is plausible to envisage more sophisticated risk evaluations explicitly taking into account the co-presence of other datasets in the data environment. In harness with web crawling software it is even possible to envisage comprehensive data environment analyses.

## References


Blumofe, R, Joerg, C., Kuszmaul, B., Leiserson, C., Randall, K. & Zhou, Y (1995), 'Cilk: An Efficient Multithreaded Runtime System' In *Proceedings of the 5th Symposium on Principles and Practice of Parallel Programming*, 207-21, Santa Barbara, Calif.

Desell, T. and El Maghraoui, K. and Varela, C. (2004) 'Load Balancing of Autonomous Actors over Dynamic Networks' In *Proceedings of the Proceedings of the 37th Annual Hawaii International Conference on System Sciences*, HICSS'04, 90268.1, Washington, DC, USA.

Elliot, M, J, (1999). 'DIS: Data Intrusion Simulation - a method of estimating the worst case disclosure risk for a microdata file'. In *Proceedings of an international symposium on linked employee-employer records.* Washington: Bureau of the Census.

Elliot, M. J., Manning, A. M.& Ford, R. W. (2002**).** 'A Computational Algorithm for Handling the Special Uniques Problem'. *International Journal of Uncertainty, Fuzziness and Knowledge Based Systems* 5(10), pp 493-509.

Elliot, M. J., Skinner, C. J., and Dale, A. (1998). 'Special Uniques, Random Uniques, and Sticky Populations: Some Counterintuitive Effects of Geographical Detail on Disclosure Risk'. *Research in Official Statistics* 1(2), pp 53-67.

Fienberg, S. E. and Makov M. M. (1998) Confidentiality, Uniqueness and Disclosure Limitation for Categorical Data, *Journal of Official Statistics* **14 (4)**, 395-98.

Goux, J-P and Kulkarni, S., Yoder, M. & Linderoth, J.,(1995) 'Master\\u2013Worker: An Enabling Framework for Applications on the Computational Grid' *Cluster Computing*, 4(1), 63—70.

Kee, Y. & Ha, S(1998) 'A Robust Dynamic Load-balancing Scheme for Data Parallel Application on Multicomputer Systems' In *Proceedings of International Conference on Parallel and Distributed Processing Techniques and Applications*, 974-980, Las Vagas, USA.

Manning A. M. and Haglin, D. J. (2005) "A new algorithm for finding Minimal Sample Uniques for use in Statistical Disclosure Assessment", *Proceedings of The Fifth IEEE International Conference on Data Mining*, New Orleans, Louisiana, U.S.A., November 27-30, 2005.

Mayes, K.R., Lujan, M Riley, G.D. Chin, J., Coveney P.V., & Gurd, J.R. (2005) 'Towards Performance Control on the Grid' Philosophical Transactions of the Royal Society of London Series A, 363 (1833), 1793-1805.

Skinner, C. J. and Elliot, M. J. (2002). 'A measure of disclosure risk for microdata', *Journal of the Royal Statistical Society Series B*, 64(4) pp 855-867.

Skinner, C. J. and Holmes, D. J.(1998) Estimating the Re-identification Risk per Record in Microdata. *Journal of Official Statistics* 14 (4), 361-372.