Fuzzy-Set / Qualitative Comparative Analysis
ユーザーガイド

Charles C. Ragin
Department of Sociology
University of Arizona
Tucson, AZ 85721
cragin@email.arizona.edu

Assisted by:

Sarah Ilene Strand
Claude Rubinson

September 2008

Based on: fsQCA 2.0

日本語訳:

森 大輔
東京大学法学政治学研究科助教

日本語版について

この日本語版マニュアルは、USER'S GUIDE TO Fuzzy-Set / Qualitative Comparative Analysis (http://www.u.arizona.edu/~cragin/fsQCA/)を日本語に翻訳したものである。翻訳は森大輔（東京大学法学政治学研究科助教）が行った。

fs/QCA は、質的比較分析（QCA）のためのソフトウェアであり、特に新しい手法であるファジィ集合質的比較分析（fsQCA）を行えることが大きな特徴となっている。また、旧来からのクリスプ集合質的比較分析（csQCA）についても、いくつかの新たな指標を取り入れるなど、最新の議論に対応している。

このマニュアルは、QCA の手法自体の説明にも多くのページを割いている結果、100 ページ近くの大部なものとなっている。そのため、このマニュアルをそのまま QCA に対する入門書としても利用するという手もある。

マニュアルの記述のうち、分かりにくいと思われる箇所や最新のバージョンに対応していない箇所については訳注を付してある。

最後に、日本語版の作成を快く認めてくださり、マニュアルの内容に対する質問にも E メールで懇切丁寧に答えてくださった Charles Ragin 教授に感謝の意を表したい。

なお、この日本語版へのご質問、誤りのご指摘などは、mdai@ad.wakwak.com までご連絡いただきたい。

森　大輔

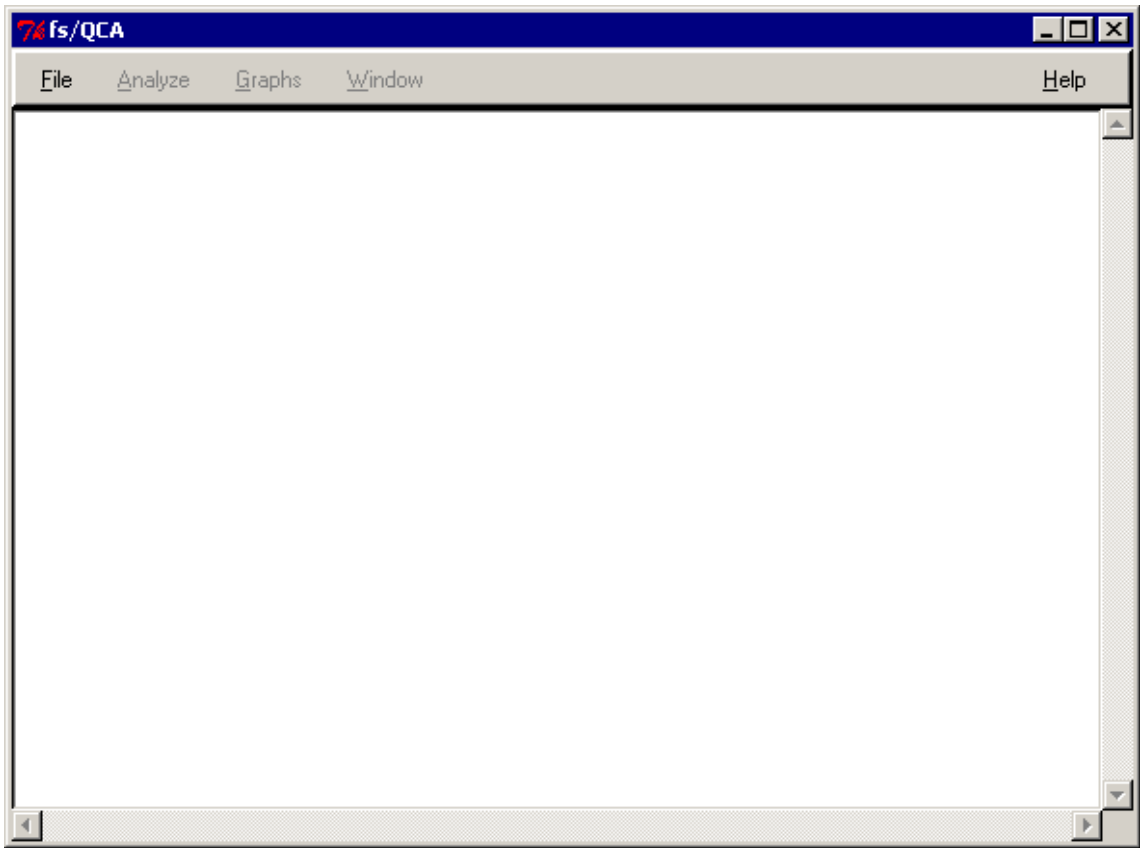
2010 年 6 月

目次

1. データファイル

A) データファイルを開く

- fsQCA を起動すると、次のようなウィンドウが開く。



- メニューから次のように選択する。
  File（ファイル）
  Open（開く）
  Data（データ）

- Open File（ファイルを開く）ダイアログボックスで、開きたいファイルを選択する。

- Open（開く）をクリックする。

B) 様々なファイル形式のデータファイルを開く

データファイルにはたくさんの種類のファイル形式がありうるが、fsQCA では次のようなファイル形式に対応している。

- QCA（*.qdm と*.qvn）　これらのファイル形式は、データを QCA3.0(DOS)に入力した際に作成される。.QDM はデータファイルで、.QVN は変数名のリストを含むファイルである。クリスプ集合のデータのみ、このフォーマットで入力したり保存したりできる。これは「古い」フォーマットであり、まもなく廃止される。

- カンマ区切り（*.csv）　カンマで区切られたファイルで、Excel などの表計算ソフトで作成される。

- スペース区切り（*.txt）　スペースで区切られたファイル。Word などのワープロソフトで作成できるが、テキスト形式で保存すること。

- タブ区切り（*.dat）　タブで区切られたファイルで、SPSS などの統計ソフトパッケージで作成できる。

推奨ファイル形式は、*.csv（Excel）と*.dat（SPSS）である。

fsQCA は、*.csv, *.dat, *.txt のデータファイルの構造について、次のことを前提としているので注意すること。第 1 に、これが最も重要なことであるが、スプレッドシートの最初の行のセルにはそれぞれの列の変数名が入る。第 2 に、データはスプレッドシートの 2 行目から始まり、それぞれの行に 1 つの事例が入る。最後に、それぞれの列のセルには同じタイプのデータが入る。データのタイプは列ごとに異なっていてもよいが、列内では同じでなければならない。句読記号やスペースを含まないアルファベット文字のみを使い、できるだけ単純な変数名にすること。例えば、"GNP1990"はよいが、"GNP 1990"や"GNP-1990"は認められない。

➢ **Excel** で作成されたデータを開く・保存する

Excel ファイルを CSV 形式（カンマ区切り）で保存する。Excel のスプレッドシートの最初の行に、変数名が入っていることを確認しておくこと。fsQCA で開く。

➢ **SPSS** で作成されたデータを開く・保存する

SPSS ファイルを DAT 形式（タブ区切り）で保存する。SPSS で保存する際、「スプレッドシートへの変数名の書き込み」というオプションがあるが、このオプションのチェックを外さないこと。

➢ **Word** や**メモ帳**で作成されたデータを開く・保存する

スペース区切りでデータを入力する。表の最初の行には、スペース区切りで変数名を入れること。TXT（書式なし）, TXT（強制改行）, TXT（MS-DOS）, TXT（MS-DOS で強制改行つき）のいずれかの形式で保存する。fsQCA で開く。

➢ 他のプログラムやファイル形式で作成されたデータを開く・保存する

STAT TRANSFER(Version 6 以降)を使うことで、様々なファイル形式のデータを CSV（カンマ区切り）ファイルに変換することができる。その後、fsQCA で開く。

プログラム	拡張子	プログラム	拡張子
1-2-3	WK*	OSIRIS	DICT
Access	MDB	Paradox	DB
DBASE や互換ソフト	DBF	Quattro Pro	WQ*, WB*
Epi Info	REC	SAS の Windows 版と OS/2 版	SD2
EXCEL	XLS	SAS の Unix 版	SSD*
FoxPro	DBF	SAS 移送ファイル	XPT
Gauss	DAT	S-PLUS	なし
JMP	JMP	SPSS データファイル	SAV
LIMDEP	CPJ	SPSS ポータブルファイル	POR
Matlab	MAT	Stata	DTA
Mineset	SCHEMA	Statistica	STA
Minitab	MTW	SYSTAT	SYS
ODBC データソース	なし		

→ ASCII-Delimited 形式の TXT か CSV に変換する

C) ファイルの保存オプション

➢ メニューから次のように選択する。

File（ファイル）

Save（保存）

- 同じ名前で同じ箇所に存在している前のバージョンのファイルが上書きされ、修正されたファイルが保存される。

または

- 新しいデータファイルを保存したり、データを違うファイル形式で保存したりするには、メニューから次のように選択する。
  File（ファイル）
    Save As（名前を付けて保存）

- ドロップダウンリストからファイル形式を選択する。

- 新しいデータファイルのファイル名を入力する。*.csv か*.dat を使うことを推奨する。

D) fsQCA のデータを他のファイル形式で開く

fsQCA にデータを入れていくらかの予備的な分析を終えた後、データを fsQCA で編集することもできるし（第 2 章参照）、自分が慣れたソフトウェアパッケージ（例えば SPSS や Excel）の助けを借りて編集することもできる。同様に、fsQCA でデータをグラフィック表示することもできるし（第 3 章表示）、SPSS や Excel のより洗練されたグラフィック表示を利用することもできる。もし SPSS や Excel でこれらの操作をすることを選択する場合、fsQCA でファイルを保存して、自分の選択したプログラムに転送する必要がある。

SPSS

- SPSS で fsQCA のデータを開くには、fsQCA のデータスプレッドシートをタブ区切りのファイル形式(*.dat)で保存する。fsQCA のデータファイルの変数の文字列は、間に空白がないように書かなくてはならないことに注意する（空白を含む変数は使えない）。

- SPSS で次のように選択する。
  ファイル
    テキストデータの読み込み

- 保存した fsQCA のファイルを開く。

- SPSS がファイルについていくつか質問をしてくる。次のオプションをチェックする。

テキストファイルは定義済みの形式に一致しますか？	いいえ
元データの形式	自由書式
ファイルの先頭に変数名を含んでいますか？	はい
最初の事例の取り込み開始行番号	2
ケースの表される方法	各行が 1 つのケースを表す
インポートするケース数	すべてのケース
変数間に使用する区切り記号	タブ
テキスト修飾子	なし
あとで使用できるようにこのファイル形式を保存しますか？	はい/いいえ
シンタックスを貼り付けますか？	いいえ

これらの後、完了をクリックする

- これで SPSS でデータを編集したりグラフィック表示を行ったりできるようになる。

- SPSS ファイルを fsQCA に再び転送する方法は、第 1 章 B の SPSS の項を参照。

Excel

- **Excel** で fsQCA のデータを開くには、fsQCA のデータスプレッドシートをカンマ区切りのファイル形式(*.csv)で保存する。fsQCA のデータファイルの変数の文字列は、間に空白がないように書かなくてはならないことに注意する（空白を含む変数は使えない)。

- Excel で次のように選択する。
  ファイル
  　開く

- 保存した fsQCA のファイルを開く。

- これで Excel でデータを編集したりグラフィック表示を行ったりできるようになる。

- Excel ファイルを fsQCA に再び転送する方法は、上記を参照。

2. データエディタ

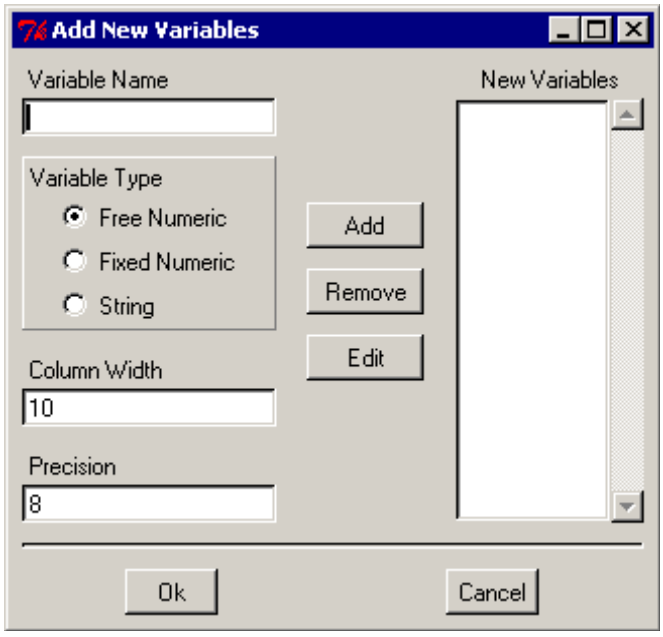
A) データの入力（最初から fsQCA でデータファイルを作成する）

➢ メニューから次のように選択する。

File（ファイル）

New（新規作成）

➢ Add New Variables（新しい変数の追加）ウィンドウが開く。



➢ **Variable Name**（変数名）を入力する。変数名には次のようなルールが適用される。

- 名前の長さは 15 文字を越えることはできない。
- それぞれの変数名は 1 回のみ使える。同じ変数名を 2 回使用することはできない。
- 変数名は大文字小文字の区別をできない。NEWVAR と NewVAR と newvar という名前は全て同じと見なされる。
- 変数名には空白やハイフンを含んではならない。
- 半角英数文字のみが使用できる（0-9, a-Z)。

➢ ３つの可能な **Variable Type**（変数のタイプ）から選択する。デフォルトでは、全ての新しい変数は、Free Numeric（浮動小数点数）とされている。

- Free Numeric（浮動小数点数）　このタイプを選択して入力されたデータは、どんな精度ともなりうる（下記の小数点以下の桁数の制限以内で)。
- Fixed Numeric（固定小数点数）　このタイプのデータは、指示された小数点以下の桁数に常に固定される。
- String（文字列）

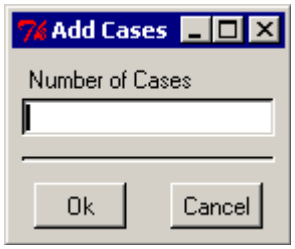
➢ **Column Width**（列の幅）を決める。このオプションはデータシートの列の幅を決める。

- **Precision**（精度）の水準を決定する。このオプションで小数点以下の桁数をコントロールできる。

- Add（追加）ボタンをクリックすると、New Variable（新しい変数）のリストに変数が追加される。

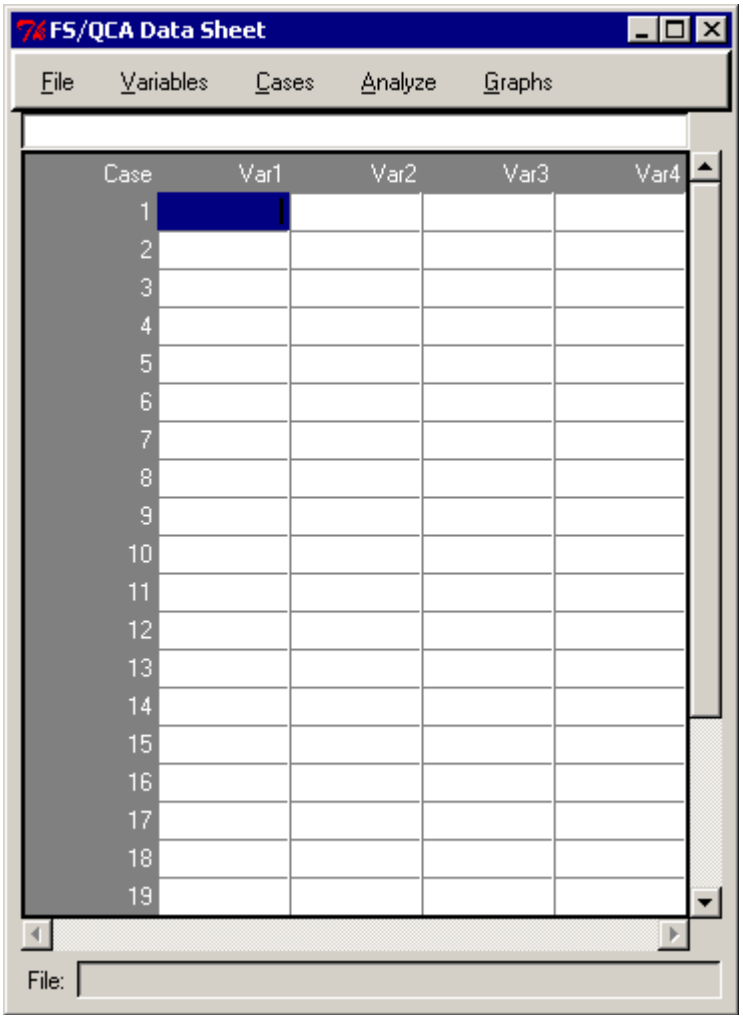
- また、New Variable（新しい変数）のリストに既に追加した変数をハイライト表示して Remove（削除）ボタンをクリックすることで、その変数を削除することができる。同様に、既に追加した変数のタイプ、列の幅、精度を変更する必要がある場合は、その変数をハイライト表示して Edit（編集）ボタンをクリックすればよい。

- 全ての変数を入力したら Ok をクリックする。新しいウィンドウが開き、データセットの **Number of Cases**（事例の数）を聞いてくる。



注: 一般的には、fsQCA は多数のデータを処理することが可能である。しかし、fsQCA の重要な特徴は、原因の組み合わせを扱うことにある。変数を追加することは事例を追加することよりもずっと計算時間に影響を与える。k を原因条件の数だとすれば、可能な組み合わせの数は 2 の k 乗である。目安として、10 以下の原因条件（1024 の可能な組み合わせがある）は問題ない。10 以上の条件を扱うときは、プログラムが分析を行うのを待つ必要が出てくる。

- データセットの事例の数を入力し、Ok ボタンを押す。すると、次のような Data Sheet（データシート）ウィンドウが開く。



- **データの値を入力する。**どのような順序で入力してもよい。選択された範囲または個々のセルについて、事例ごとに入力してもよいし、変数ごとに入力してもよい。アクティブなセルは濃い色でハイライトされる。セルを選択しデータの値を入力すると、値がメニューバーの下のセルエディタに表示される。値は数でも文字列でもよい。データの値は Enter キーを押すまでは記録されない。

- データシートを閉じる前に、入力された情報が失われないように保存する必要がある。

B) データの編集

変数の追加・削除

- 既存のデータシートに**変数を追加する**には、次のように選択する。
  Variables（変数）

Add（追加）

- 変数とその詳細を入力したら Add（追加）を押す。

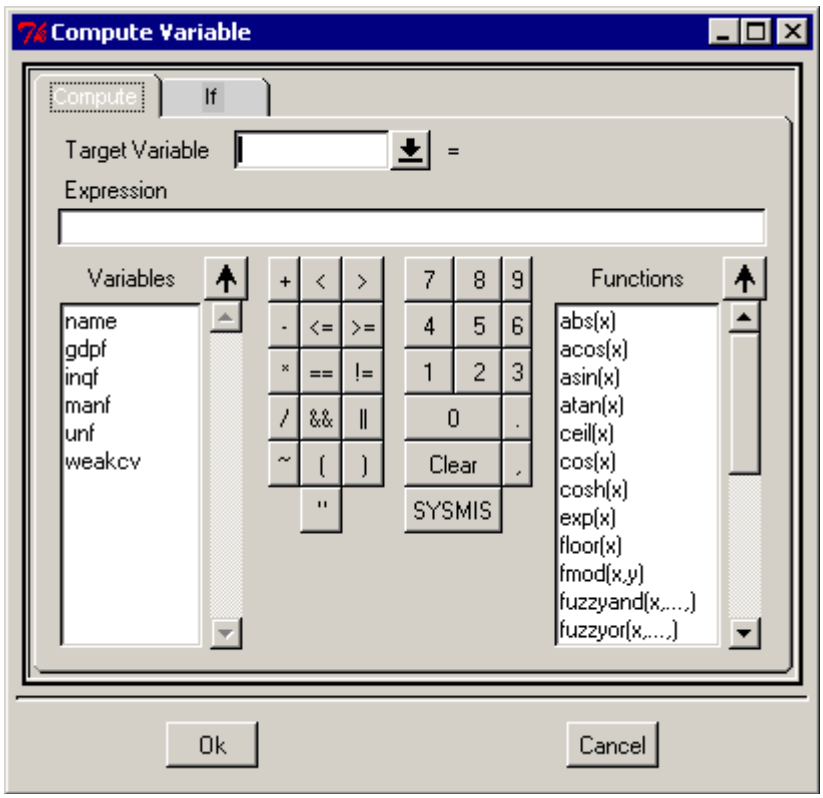
- データシートから**既存の変数を削除する**には、次のように選択する。
  Variables（変数）
    Delete（削除）

- 削除したい変数をハイライトして、Delete Variable?（変数を削除しますか?）で Ok をクリックする。

変数の計算

- 既存の変数や数式、論理式などから**新しい変数を計算する**には、次のように選択する:
  Variables（変数）
    Compute（計算）

- 次のようなウィンドウが開く（データファイルの変数名がウィンドウにリスト化される）



- Target Variable（目的変数）の名前をタイプする。目的変数は既存の変数でもよいし、現在のデータファイルに付け加える新しい変数でもよい。変数名は 1 文字（例えば、"X")にしてはならない。クラッシュする原因となる。

➢ 式を作るには、Expression（式）のフィールドにペーストするか直接入力すればよい。

1）算術演算子

+ **加算**。前の項と後ろの項を足す。両方の項が数でなければならない。

- **減算**。前の項から後ろの項を引く。両方の項が数でなければならない。

* **乗算**。前の項と後ろの項を掛ける。両方の項が数でなければならない。

/ **除算**。前の項を後ろの項で割る。両方の項が数でなければならず、後ろの項は 0 であってはならない。

2）関係演算子

< **より小さい**。数の場合、前の項が後ろの項よりも小さいとき、真（=1）を返す。文字列の場合、前の項が後ろの項よりも照合順序（アルファベット順）で早く現れるものであるならば、真を返す。この演算子は普通、論理条件でしか使われない。

> **より大きい**。数の場合、前の項が後ろの項よりも大きいとき、真（=1）を返す。文字列の場合、前の項が後ろの項よりも照合順序（アルファベット順）で後に現れるものであるならば、真を返す。この演算子は普通、論理条件でしか使われない。

<= **以下**。数の場合、前の項が後ろの項よりも小さいか等しいとき、真（=1）を返す。文字列の場合、前の項が後ろの項よりも照合順序（アルファベット順）で早く現れるものであるか、２つの項が同じならば、真を返す。この演算子は普通、論理条件でしか使われない。

>= **以上**。数の場合、前の項が後ろの項よりも大きいか等しいとき、真（=1）を返す。文字列の場合、前の項が後ろの項よりも照合順序（アルファベット順）で後に現れるものであるか、２つの項が同じならば、真を返す。この演算子は普通、論理条件でしか使われない。

== **等号**。両項が完全に等しいとき、真（=1）を返す。文字列の項の長さが等しくないときは、比較する前に短い方の項の右側に空白が付加される。この演算子

は普通、論理条件でしか使われない。

!= **等号否定**。両項が完全には等しくないとき、真（=1）を返す。文字列の項の長さが等しくないときは、比較する前に短い方の項の右側に空白が付加される。この演算子は普通、論理条件でしか使われない。

&& **論理積（「かつ」）**。前の項と後ろの項の両方が真であるとき、真（=1）を返す。両項は論理項でも数の項でもよい。0 より大きい数の項は真と扱われる。この演算子は普通、論理条件でしか使われない。

| | **論理和（「または」）**。前の項か後ろの項が真であるとき、真を返す。両項は論理項でも数値でもよい。0 より大きい数の項は真と扱われる。この演算子は普通、論理条件でしか使われない。この演算子は、記号を式の欄にペーストした場合にしか働かない。

~ **論理否定**。後ろの項が偽であるとき、真を返す。1 - (数値の項)。この演算子は普通、論理条件でしか使われない。

3) 算術関数

abs(x) x の絶対値を返す。x は数値でなければならない。

acos(x) x のアークコサイン（コサインの逆関数）を返す。x はラジアンで測った 0 から 1 の間の数値でなければならない。

asin(x) x のアークサイン（サインの逆関数）を返す。x はラジアンで測った 0 から 1 の間の数値でなければならない。

atan(x) x のアークタンジェント（タンジェントの逆関数）を返す。x はラジアンで測った数値でなければならない。

ceil(x) x を切り上げした結果の整数を返す。x は数値でなければならない。
例: ceil(2.5)=3.0

calibrate 間隔尺度や比率尺度の変数をファジィ集合に変形する。詳細は後述。

cos(x) x のコサインを返す。x はラジアンで測った数値でなければならない。

cosh(x)　　x のハイパボリックコサイン［$(e^x + e^{-x})/2$］を返す。x はラジアンで測った数値でなければならない。x の値は 230 を越えることはできない。

exp(x)　　e の x 乗を返す。e は自然対数の底である。x は数値でなければならない。x の値を大きくする(x>230)と、マシンの処理能力を超えてしまう。

floor(x)　　x を切り捨てした結果の整数を返す。x は数値でなければならない。
例: floor(2.5) = 2.0

fmod(x,y)　　x を法（modulus）である y で割った剰余を返す。両方の引数とも数値でなければならず、y は 0 であってはならない。

fuzzyand(x,...,)　2 つ以上のファジィ集合の最小値を返す。
例: fuzzyand(1.0, 0.1) = 0.1

fuzzyor(x,...,)　　2 つ以上のファジィ集合の最大値を返す。
例: fuzzyor(1.0, 0.1) = 1.0

fuzzynot(x)　　ファジィ集合の否定（1-x）を返す（否定「~ 」と同じ)。
例: fuzzynot(0.8) = 0.2

int(x)　　x の整数部分を返す。数値が切り捨てされて整数になる。

log(x)　　x の e を底とする対数を返す。x は 0 より大きい数値でなければならない。

log10(x)　　x の 10 を底とする対数を返す。x は 0 より大きい数値でなければならない。

pow(x,y)　　前の項を後ろの項で累乗したものを返す。前の項が負のときは、後ろの項は整数でなければならない。特に後ろの項（指数）が非常に大きいか非常に小さい場合、この演算子の結果は大きくなりすぎたり小さくなりすぎたりして、マシンの処理能力を超える可能性がある。

round(x)　　x を四捨五入した結果の整数を返す。x は数値でなければならない。ちょうど.5 となるような数は切り上げされる。
例: round(2.5) = 3.0

sin(x)　　x のサインを返す。x はラジアンで測った数値でなければならない。

sinh(x)　　x のハイパボリックサイン［$(e^x - e^{-x})/2$］を返す。x はラジアンで測った数値でなければならない。x の値は 230 を越えることはできない。

square(x)　　x の 2 乗を返す。x は数値でなければならない。

sqrt(x)　　x の正の平方根を返す。x は負でない数値でなければならない。

tan(x)　　x のタンジェント［sin/cos］を返す。x はラジアンで測った数値でなければならない。

tanh(x)　　x のハイパボリックタンジェント［$(e^x - e^{-x})/ (e^x + e^{-x})$］を返す。x はラジアンで測った数値でなければならない。

4) その他の演算子

( )　　**グループ化**。括弧の中の演算子や関数は、括弧の外の演算子や関数よりも先に計算される。

"　　**引用符**。文字列の変数の値を示すのに用いられる。
例: 以下の IF 条件を満たす場合に計算: Variable == "NA"

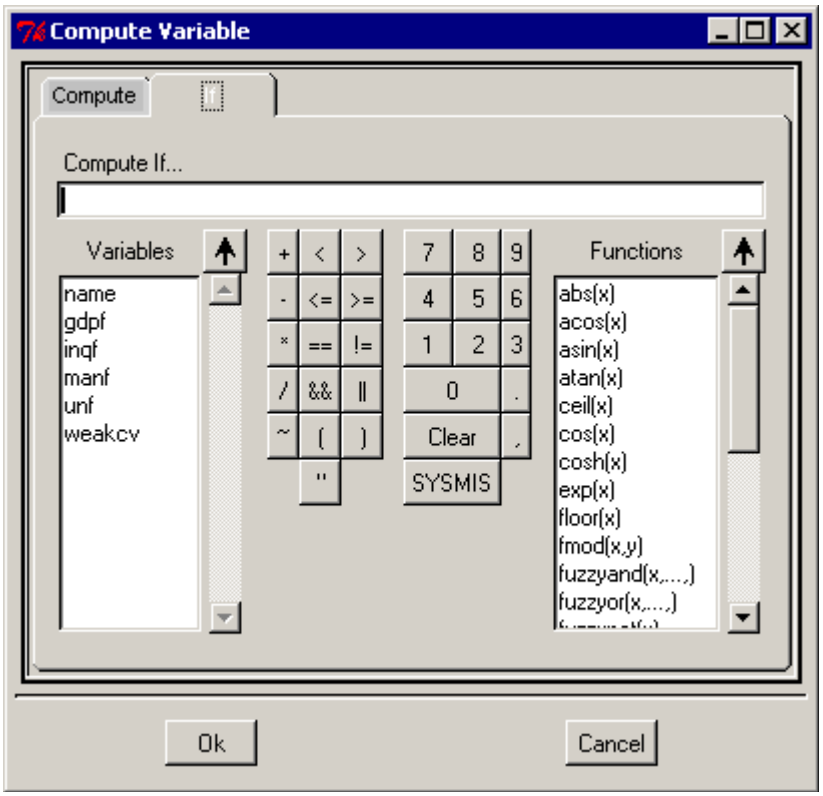
SYSMIS　　**システム標準の欠損値**。事例のサブセットを選択する際に用いられる。
例: 以下の IF 条件を満たす場合に選択: Variable == SYSMIS

Clear　　Expression（式）のフィールドの記述を**削除**する。

IF 条件を満たす場合に変数を計算

IF ダイアログボックスでは条件式を用いることで、事例のサブセットを選択してデータの変換を適用することができる。条件式は、真、偽、欠損のどれかの値をそれぞれの事例について返す。

- データの変換に関係ある事例を選択するには、以下のように選択する。
  Variables（変数）
  　Compute(計算)...
- If をクリックすると、以下のようなダイアログボックスが現れる。



- 条件式の結果が真ならば、変換が事例に適用される。

- 条件式の結果が偽または欠損であるならば、変換は事例に適用されない。

- たいていの条件式は、電卓のキーパッドにある 6 つの関係演算子(<, >, <=, >=, ==, !=)のどれかを 1 つ以上使っている。

- 条件式は、変数の名前、定数、算術演算子、数値関数や他の関数、論理変数、関係演算子を含みうる。

変数の再コーディング

再コーディングすることでデータの値を修正することができる。これは、複数のカテゴリーを結合させるときに特に有用である。既存の変数を上書きして再コーディングすることもできるし、既存の変数に記録された値をもとに新しい変数を作ることもできる。

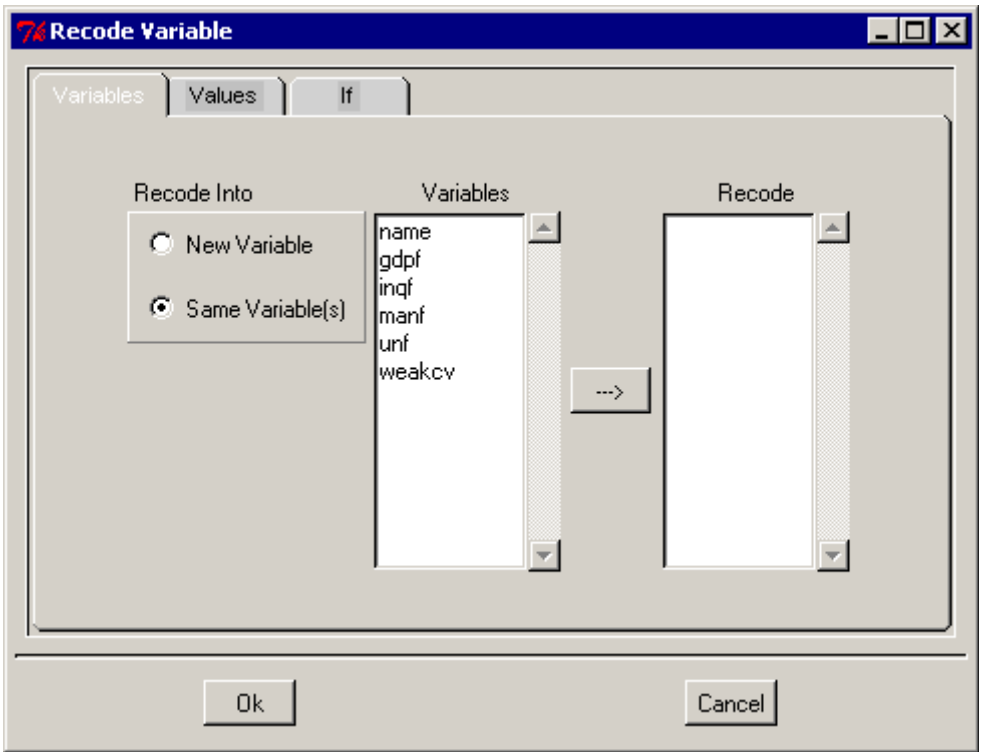
1) 同じ変数に再コーディング

既存の変数の値の再割り当てを行ったり、既存の値を結合して新しい値にしたりすることができる。数値や文字列の変数を再コーディングすることができる。1 つまたは複数の変数を再コーデ

ィングすることができ、それらは同じタイプの変数である必要もない。数値の変数と文字列の変数を一緒に再コーディングすることができる。

- 変数の値を再コーディングするためには、次のように選択する:
  Variables（変数）
    Recode（再コーディング）

- 次のようなウィンドウが開く。

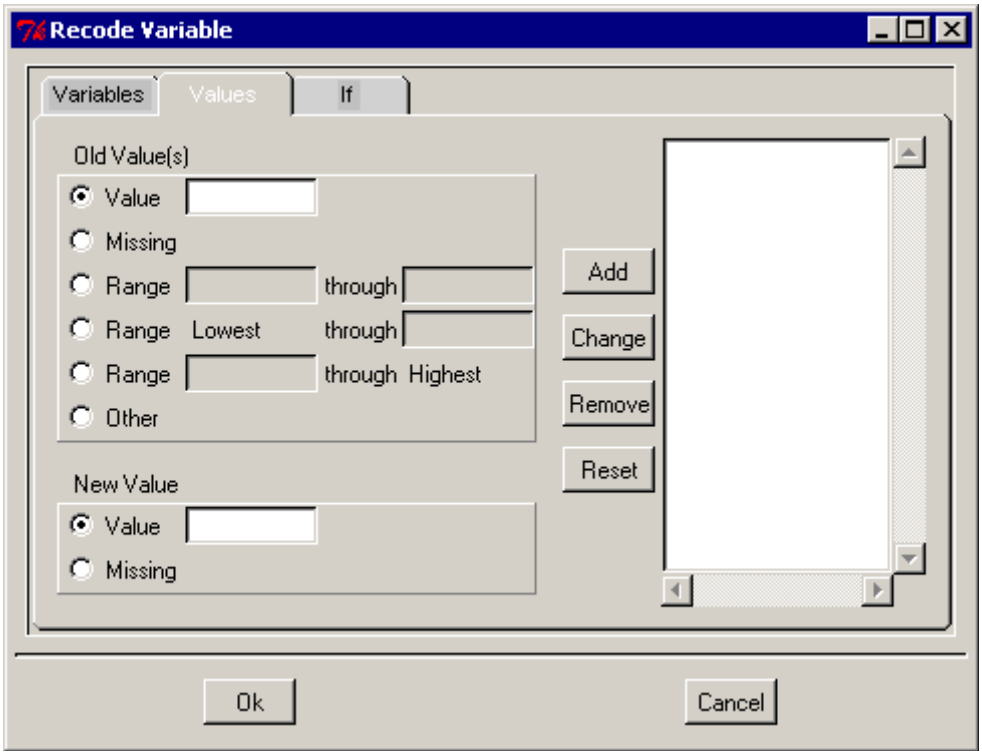


- Same Variable(s)（同じ変数）のオプションを選択する。

- 再コーディングしたい変数（数値または文字列）を選択する。

- オプションとして、再コーディングする事例のサブセットを定義することができる。事例のサブセットを定義する If ダイアログボックスは、「IF 条件を満たす場合に変数を計算」のところで説明したものと同じである。

- Values（値）をクリックして、値を再コーディングする方法を指定する。次のようなダイアログボックスが現れる。



➢ このダイアログボックスで、再コーディングする値を定義できる。

Old Value(s)（古い値） 再コーディングする前の値。単一の値、ある範囲（range）の値、欠損値を再コーディングできる。文字列の変数の場合は、範囲の概念が適用できないので、範囲は選択できない。範囲には、両端の点や範囲内にあるユーザー定義の欠損値（訳注: システム標準の欠損値ではなく、ユーザーが自分で定義した欠損値のこと）も含まれる。

New Value（新しい値） 古い単一の値やある範囲の値を再コーディングした後の値。単一の値や欠損値を割り振ることができる。

➢ Add（追加）ボタンを押すと、指定したものを右のリストに追加することができる。オプションとして、Remove（既に追加したものを取り除く）や Change（既に追加したものを変更する）もある。リストを全て削除するには Reset（リセット）ボタンを使う。

2) 別の変数への再コーディング

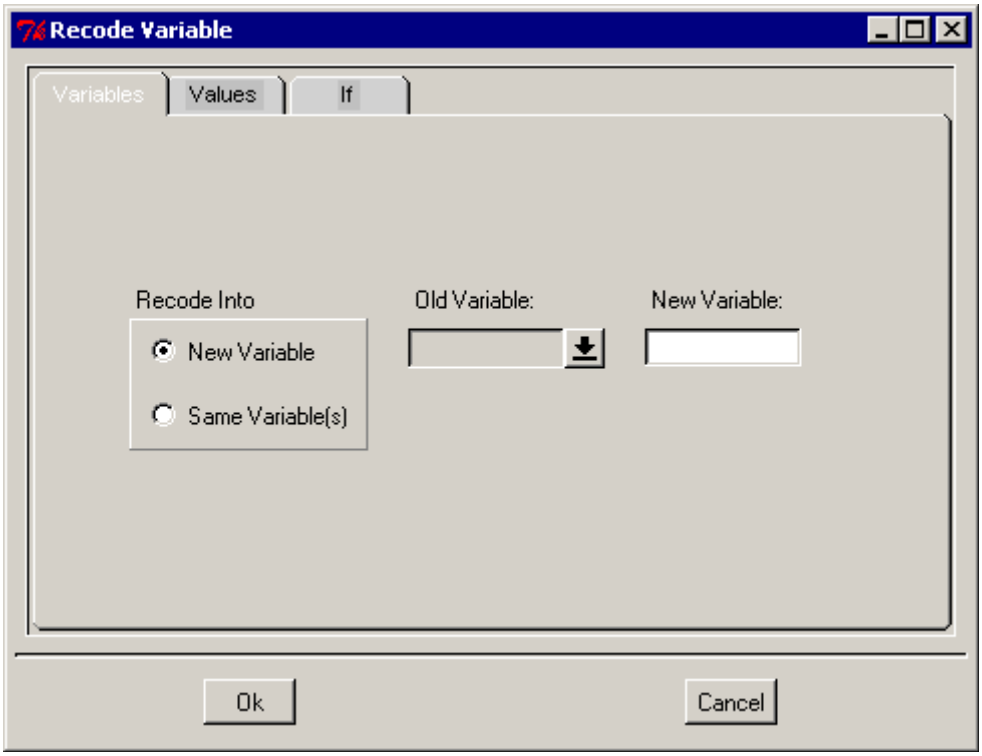
既存の変数の値の再割り当てを行ったり、既存の値を結合して新しい値にしたりして、新しい変数にすることができる。

- 数値や文字列の変数を再コーディングできる。

- 数値の変数を文字列の変数に再コーディングしたり、その逆のことをしたりできる。

- 古い変数の値を新しい変数に再コーディングするには、次のように選択する。
  Variables（変数）
  Recode（再コーディング）
  New Variable（新しい変数）

- 次のようなウィンドウが現れる。



- 再コーディングする（古い）変数をドロップダウンメニューから選択するか、フィールドにタイプする。

- （新しい）出力変数の名前を入力する。

- Values（値）タブをクリックして、値の再コーディングの方法を指定する。

- オプションとして、再コーディングする事例のサブセットを定義することができる。事例のサブセットを定義する If ダイアログボックスは、「IF 条件を満たす場合に変数を計算」のところで説明したものと同じである。

ファジィ集合のキャリブレーション

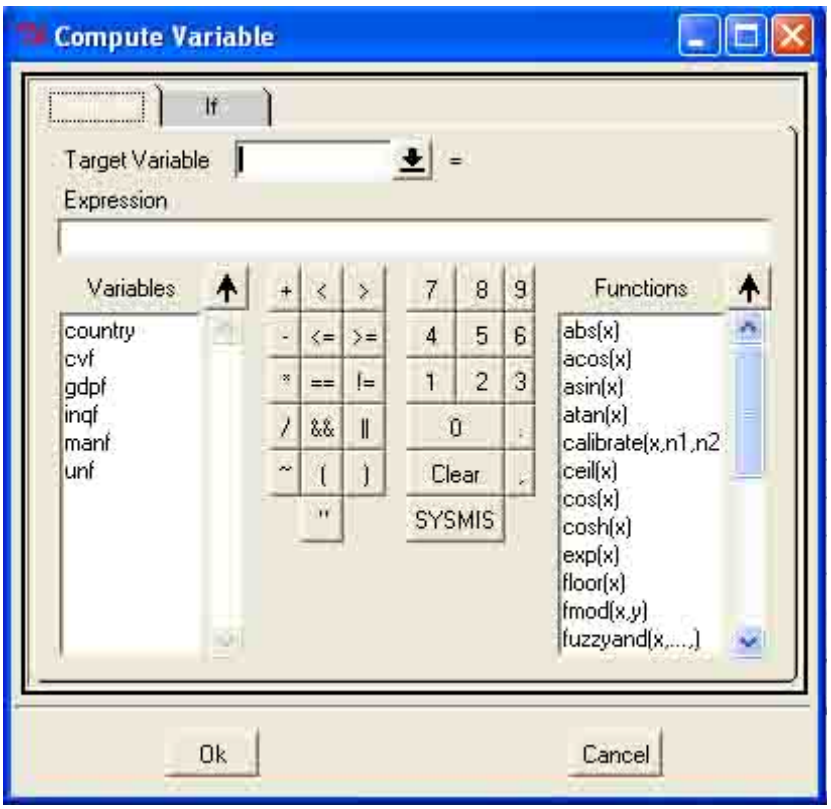
通常の変数をファジィ変数に変換するには、外から与えられた基準に変数が合致するようにするキャリブレーション（calibration）が必要である。たいていの社会科学者は、事例同士の相対的な位置関係を示すだけのキャリブレーションされていない測定値を用いることで満足している。しかし、キャリブレーションされていない測定値は、キャリブレーションされた測定値に明らかに劣る。例えば、民主主義についてのキャリブレーションされていない測定値を使うと、ある国が他の国よりも民主的であることや平均よりも民主的であることはわかるが、その国がそもそも民主的なのか独裁的なのかということが分からない。

ファジィ集合においては、データ外の理論的・実質的基準を用いてキャリブレーションが行われ、問題となっている集合についての研究者による概念化・定義・ラベリングが考慮される。最終的に、0.0 から 1.0 の値を取るそれぞれの事例のメンバーシップ度についての、きめ細かいキャリブレーションがなされる。

研究者は、ファジィ集合を構成する質的な区切り点（breakpoint）として、次の 3 つの点の間隔尺度変数の値を特定しなければならない。すなわち、完全帰属（full membership）の閾値(メンバーシップ度=0.95)、完全非帰属（full nonmembership）の閾値（メンバーシップ度=0.05)、分岐点（cross-over point）（メンバーシップ度=0.5）の 3 つである。これら 3 つの基準点は、もとの比率尺度や間隔尺度の値をファジィ集合のメンバーシップ度に変換するために使われる。変換は、完全帰属の対数オッズをもとにして行われる。

- メニューから次のように選択する。
  Variables（変数）
    Compute（計算）

- 次のようなウィンドウが開く。



- Target Variable（目的変数）の名前をつける（2～8 文字の半角英数文字を使い、空白、ダッシュ、句読点を含んではならない)。

- Functions（関数）メニューの calibrate(x,n1,n2,n3)をクリックする。Functions という文字の横にある上向きの矢印をクリックする。

- calibrate(,,,)を、例えば calibrate(oldvar,25,10,2)というふうに編集する。oldvar はファイルに既に存在する間隔尺度・比率尺度変数の名前であり、1 番目の数字は完全帰属の閾値（0.95）にあたる oldvar の値を表し、2 番目の数字は分岐点（0.5）にあたる oldvar の値を表し、3 番目の数字は完全非帰属の閾値（0.05）にあたる oldvar の値を表す。

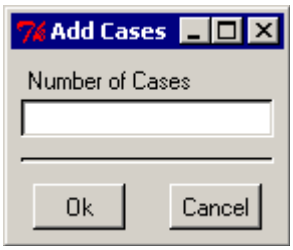
- Ok をクリックする。

- メンバーシップ度ともとの値が意図されたとおりに対応しているかをデータスプレッドシートで確認すること。プルダウンメニューで変数を昇順や降順に並び替えるのが有用であろう。以上により、0.0 から 1.0 の値を取るそれぞれの事例のメンバーシップ度についての、きめ細かいキャリブレーションがなされる。

事例の追加・挿入

- 既存のデータシートに**事例を追加する**には、次のように選択する。
  Cases（事例）
    Add（追加）

- 次のようなウィンドウが現れる。



- 追加したい事例の数を入力する。既存のデータの最後（下部）に追加した事例が現れる。

- 既存のデータシートに**事例を挿入する**には、次のように選択する。
  Cases（事例）
    Insert（挿入）

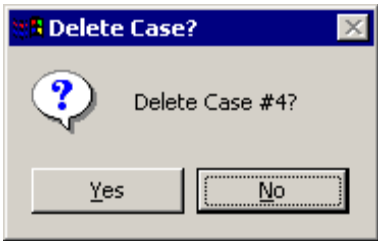
- Add（追加）ウィンドウが現れる。

- データシートのハイライトされた事例の上に挿入したい事例の数を入力する。ハイライトされたセルの事例（行）の上に、追加された事例が現れる。

事例の削除

- 既存のデータシートから **1 つの事例を削除する**には、次のように選択する。
  Cases（事例）
    Delete（削除）

- 次のようなウィンドウが現れる。



- この機能では、1 回に 1 つの事例のみ削除できる。

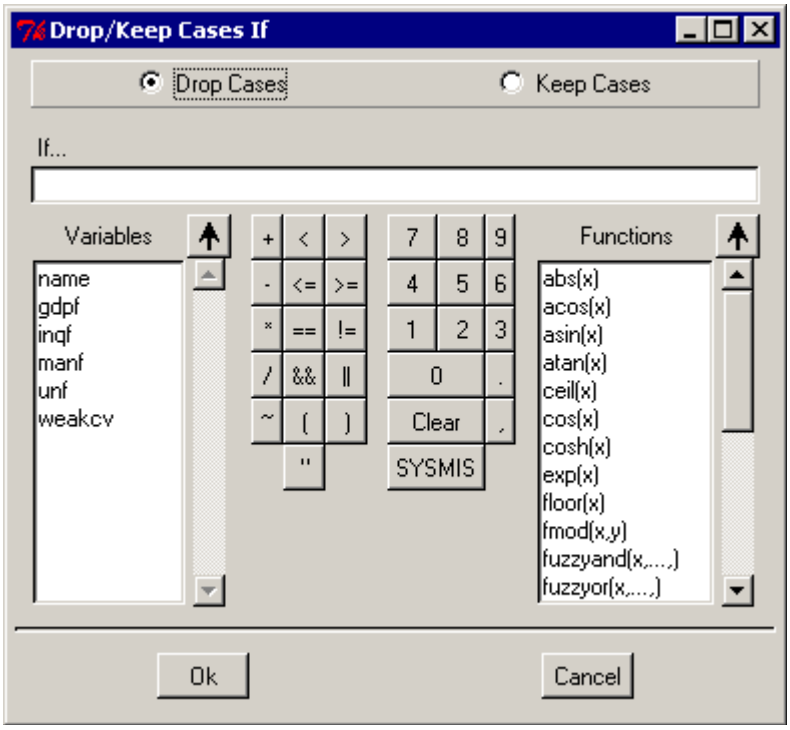
- データシートのハイライトされたセルの事例を本当に削除したいかどうか、プログラムに聞

かれる。例えば上の画像の例では、Case#4 のセルがハイライトされていた。

事例の条件つき破棄・保持

- ある特性を有する**事例を破棄したり保持したりする**には、次のように選択する。
  Cases（事例）
  Drop/Keep If（条件つき破棄・保持）

- 次のようなウィンドウが現れる。



- 保持または破棄したい事例を指定し、Ok をクリックする。

- 条件式の結果が真ならば、事例は破棄または保持される。条件式の結果が偽や欠損であるならば、事例は破棄されたり保持されたりはしない。

事例の条件つき選択

事例の条件つき選択では、以下のような変数や複合式による基準をもとにして、事例のサブグループを選択できる。

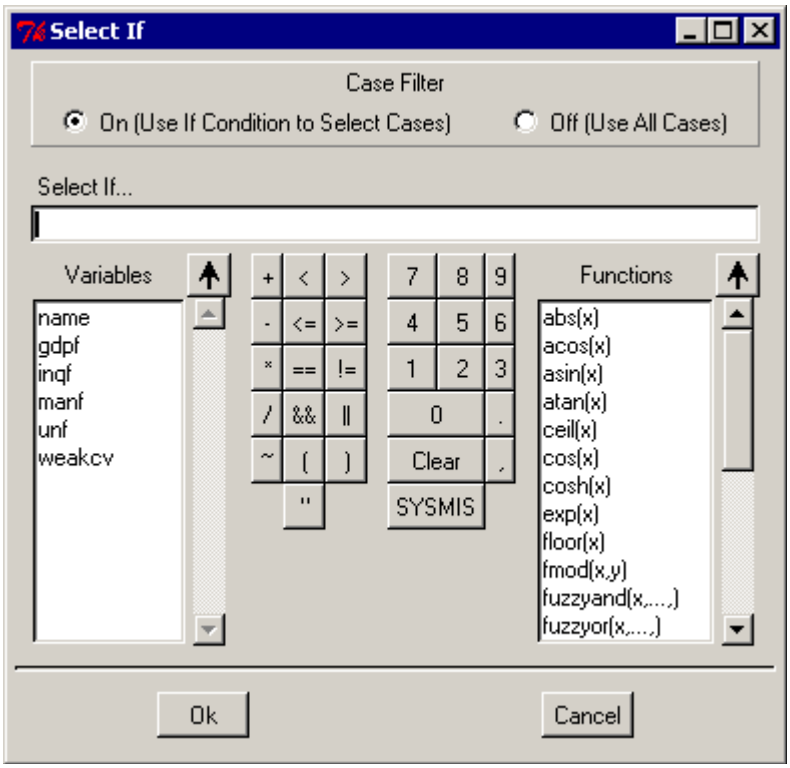
- 変数の値や範囲
- 算術式
- 論理式

- 関数

選択されなかった事例は、データファイルに残るが分析からは除外される。選択されなかった事例を示すために、データシートの行番号に括弧がつく。

- **事例のサブセットを選択して分析する**には、次のように選択する。
  Cases（事例）
  Select If（条件つき選択）

- 次のようなウィンドウが開く。



- 事例を選択する基準を指定する。

- 条件式の結果が真ならば、事例が選択される。条件式の結果が偽や欠損であるならば、事例は選択されない。

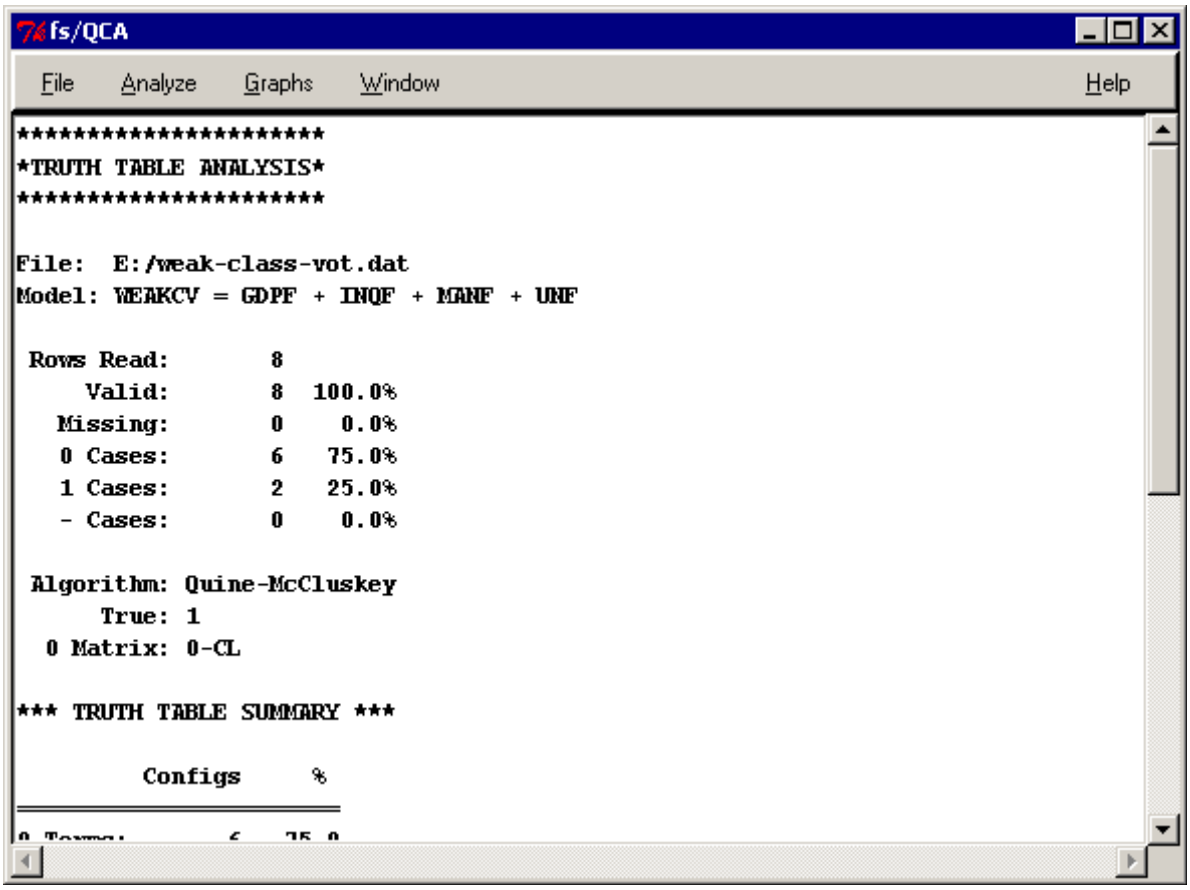
- たいていの条件式は、電卓のキーパッドにある 6 つの関係演算子(<, >, <=, >=, ==, !=)のどれか 1 つ以上を使っている。

- 条件式は、変数の名前、定数、算術演算子、数値関数や他の関数、論理変数、関係演算子を含みうる。

注: 「条件つき選択」は、単変量のときに最もうまく機能する。例えば、2 つの論理式（例:「かつ」と「または」）を組み合わせて「条件つき選択」を使いたいのであれば、その選択の基準を反映した新しい変数を（計算 compute や再コーディング recode で）作成し、その新しい変数に「条件つき選択」を使う。

C) 出力ウィンドウでの作業

プログラムの機能を実行すると、fsQCA ウィンドウに結果が表示される。スクロールバーを使って結果を閲覧することができる。



- **出力結果を印刷する**には、次のように選択する。
  File（ファイル）
  Print（印刷）

- 自分の PC で指定されているプリンタオプションのウィンドウが現れ、印刷オプションを指定することができる。

- 出力結果の他に、各ページの先頭に日時やページ番号が印刷される。

- 出力結果は、プログラム間の転送が容易なように、シングルスペースでフォントは New Courier(サイズは 10 ポイント)で記されている。したがって、SPSS などの他のプログラムで*.out ファイルを開く場合、適当なフォントに指定しなおさないと表の数字が少しずれる。

- 出力結果は、Word、Wordpad、テキストエディタなどにコピーペーストすることもできる。また、文字や簡単な表を、出力ウィンドウに直接タイプして挿入することもできる。出力ウィンドウで出力結果の一部をハイライトして、キーボードのバックスペースキーやデリートキーを押すことによって、その部分を削除することもできる。

- **出力結果を保存する**には、次のように選択する:
  File（ファイル）
    Save As（名前を付けて保存）
      Output（出力）

- fsQCA では、出力結果を保存するオプションとして、fsQCA 独自の出力ファイル（*.out）と、他のテキスト形式（例: txt）を選ぶことができる。

3. 基本統計量とグラフ

注: データスプレッドシートにおいて、いずれかの事例で変数の値が「ドント・ケア」(don't care)(ダッシュで表示）とされている場合、その変数の統計量、ヒストグラム、XY プロットは求めることができない。しかし、棒グラフは作成することができる（データスプレッドシートで「ドント・ケア」のコーディングを使用することは稀で、通常は真理表の入力を直接データスプレッドシートに移した場合にのみ生じる。）

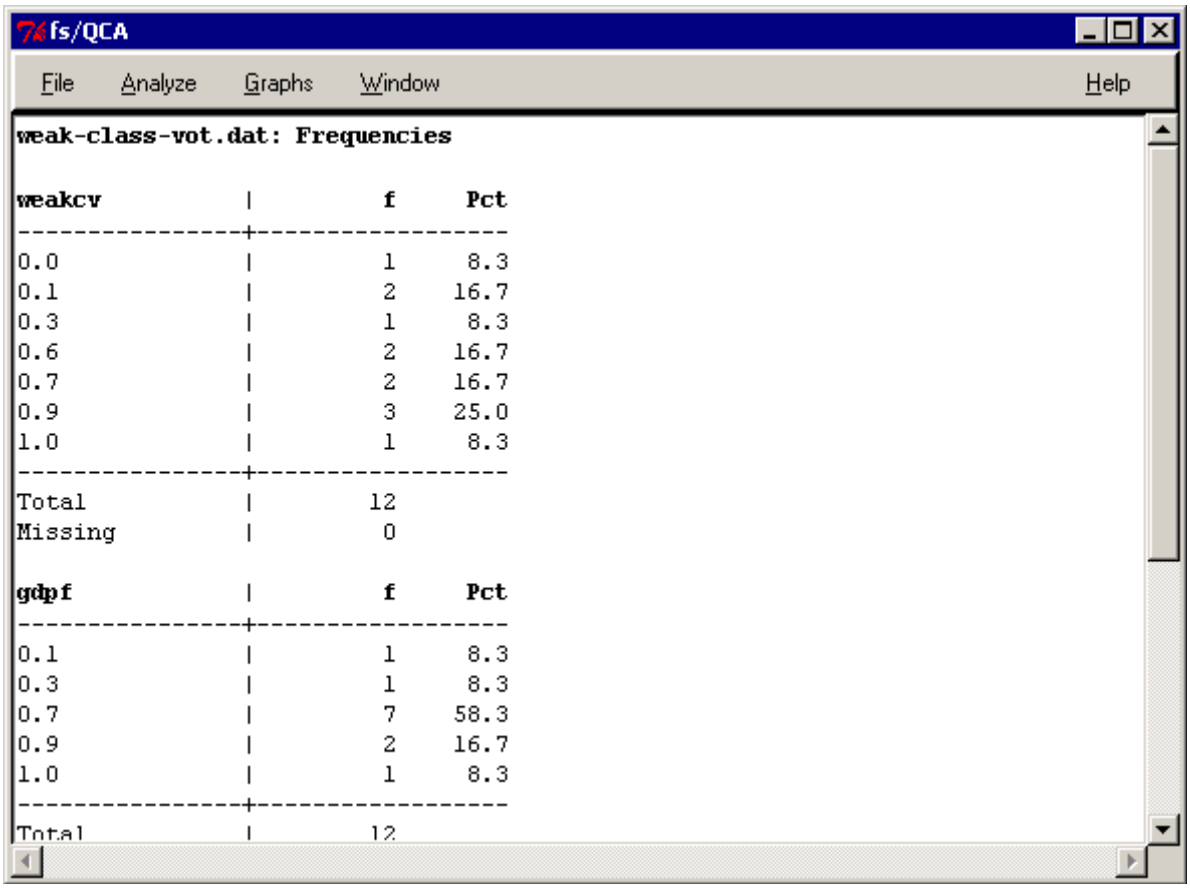
A) 度数

Frequencies（度数）という機能では、多くのタイプの変数の記述で有用な度数分布表（frequency chart）を作成できる。

- 頻度分布表を得るには、次のように選択する:
  Analyze（分析）
    Statistics（統計量）
      Frequencies（度数）

- Variables（変数）の欄から 1 つ以上の変数を選び、それを Frequencies（度数）の欄に移す。Ok をクリックする。

- 度数分布表が出力ウィンドウに現れる。



- 出力結果の最初の行には、ファイル名と選んだ機能（Frequencies）が表示される。度数分布表の列は次のものを示している。

1. 選択された変数の値
2. データセット中で該当する値の度数
3. 2 の全体に占めるパーセント（Pct）（欠損した事例は除く）

- 最後の 2 つの行は、分析した事例の総数（Total）と欠損した事例の数（Missing）を示す。

B) 記述統計量

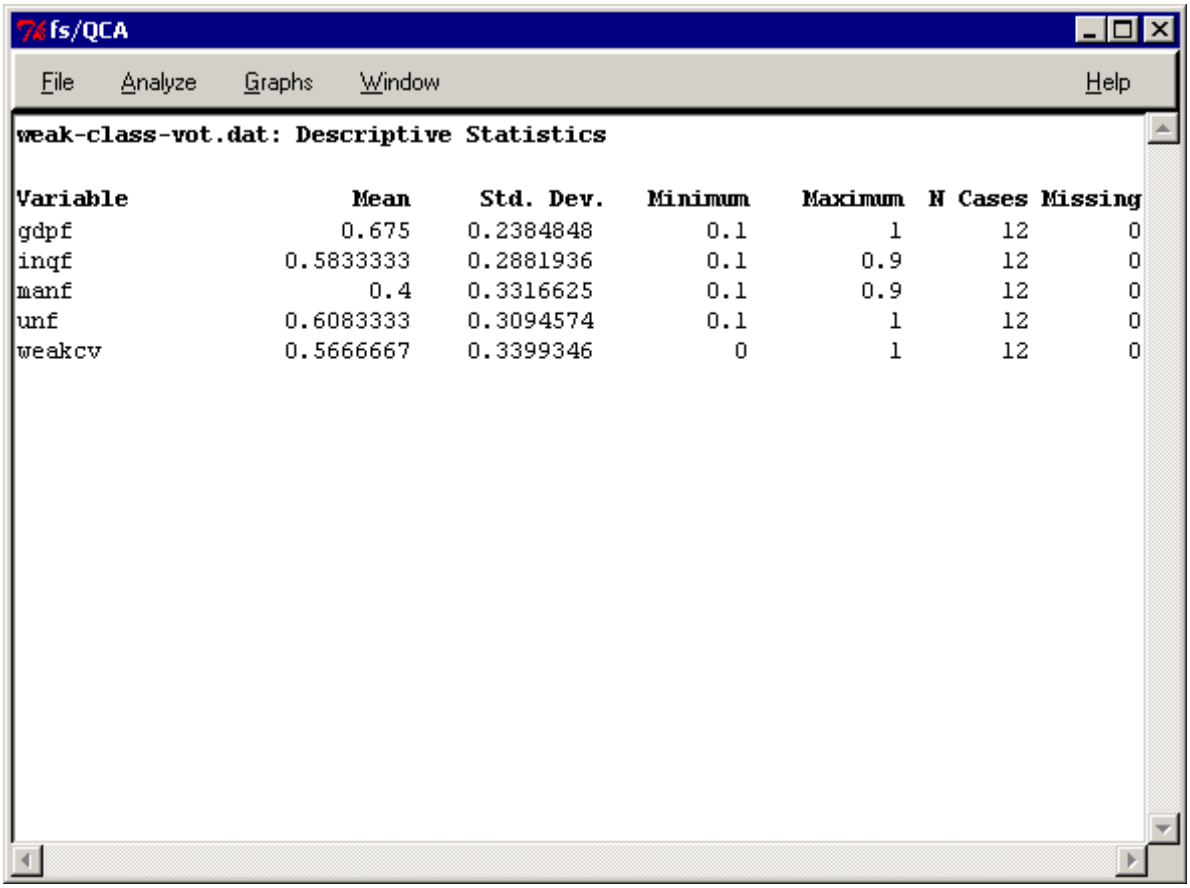
Descriptives（記述統計量）という機能では、いくつかの変数の単変量の要約統計量を単一の表

の形で表示する。

- 記述統計量を得るには、次のように選択する:
  Analyze（分析）
    Statistics（統計量）
      Descriptives（記述等計量）

- Variables（変数）の欄から 1 つ以上の変数を選び、それを Descriptives（記述等計量）の欄に移す。Ok をクリックする。

- 記述等計量が出力ウィンドウに現れる。



Variable	Mean	Std. Dev.	Minimum	Maximum	N Cases	Missing
gdpf	0.675	0.2384848	0.1	1	12	0
inqf	0.5833333	0.2881936	0.1	0.9	12	0
manf	0.4	0.3316625	0.1	0.9	12	0
unf	0.6083333	0.3094574	0.1	1	12	0
weakcv	0.5666667	0.3399346	0	1	12	0

- 出力結果の最初の行には、ファイル名と選んだ機能（Descriptive Statistics）が表示される。記述統計量の表の列は次のものを示している。

1. 選択された変数（Variable）
2. 平均値（Mean）
3. 標準偏差（Std. Dev.）

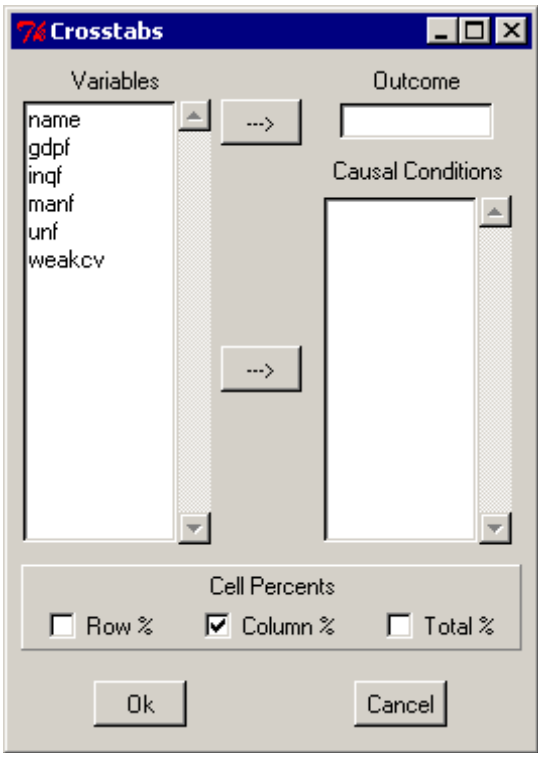
4. 変数の最小値（Minimum）
5. 変数の最大値（Maximum）
6. 事例の数（N Cases）
7. 欠損した事例の数（Missing）

C) クロス集計

Crosstabs（クロス集計）という機能では、2 次元のクロス表が作成される。多次元の表や統計量、関連の度合いなどを求める場合には、SPSS など標準的な統計パッケージを使う必要がある。

- クロス集計を得るためには、次のように選択する。
  Analyze（分析）
    Statistics（統計量）
      Crosstabs（クロス集計）

- 次のようなウィンドウが開く。



- Variables（変数）の欄から 1 つの変数を選び、それを Outcome（結果）の欄に移す。この変数は、表の行の側に現れる。

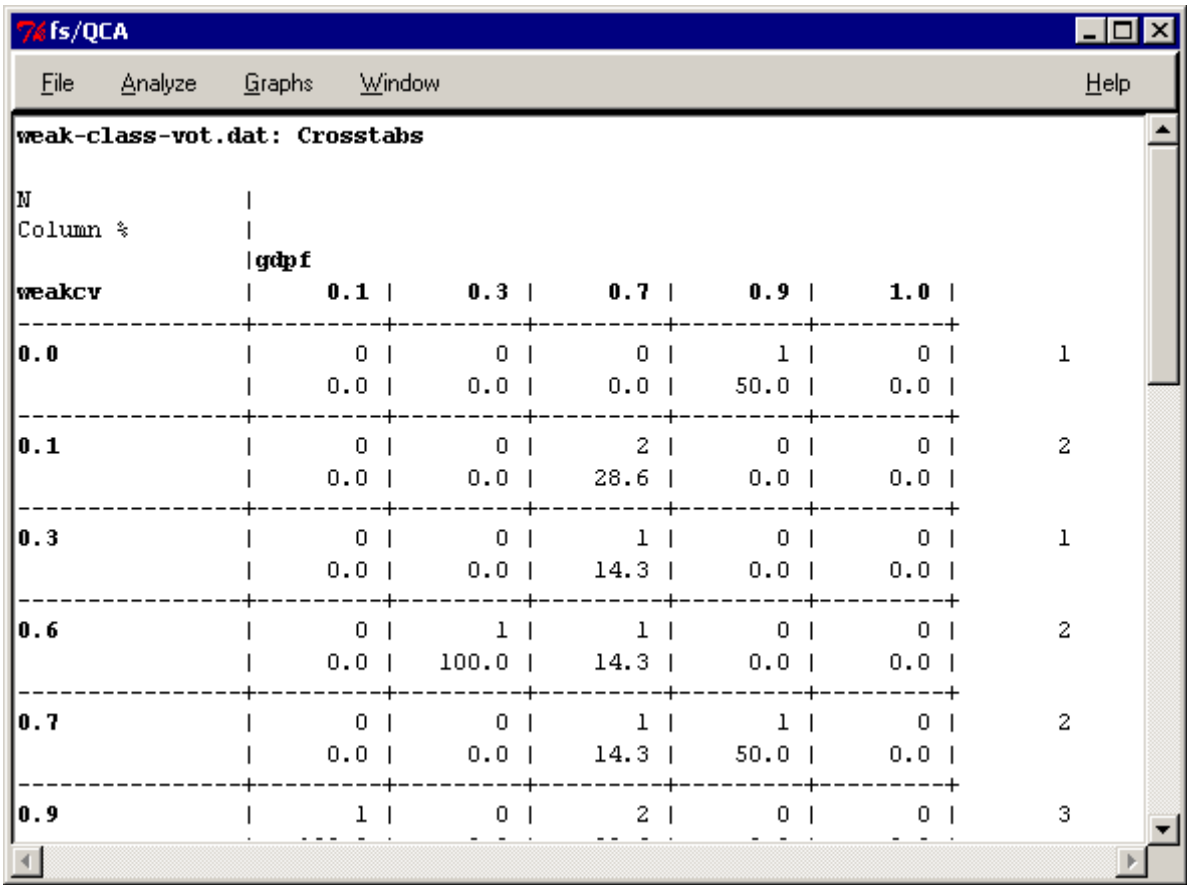
- Variables（変数）の欄から 1 つ以上の変数を選び、それを Causal Conditions（原因条件）の欄に移す。この変数は、表の列の側に現れる。

➢ 表のパーセント表示は、次のどれにするかを指定することができる。

- 当該行の総数に占めるパーセント
- 当該列の総数に占めるパーセント
- 表にある全ての事例に占めるパーセント

➢ Ok をクリックする。

➢ クロス集計が出力ウィンドウに現れる。



➢ 出力結果の最初の行には、ファイル名と選んだ機能（Crosstabs）が表示される。それぞれのセルの 1 番目の数は、そのセルに入る事例の数（N）を示している。ユーザーの指定によっては、(a) 当該行の総数に占めるパーセント（Row %）、(b) 当該列の総数に占めるパーセント（Column %）、(c) 表にある全ての事例に占めるパーセント（Total %）もいずれかも表示される。どれが表示されているかは、列ヘッダの最初のセルを見れば分かる。

➢ 表の最後の列は、それぞれの行の事例の総数を表す。

- 表の最後の行は、それぞれの列の事例の総数を表す。

- 出力結果の最後の 2 行は、分析した事例の総数（Total）と欠損した事例の数（Missing）を表す。

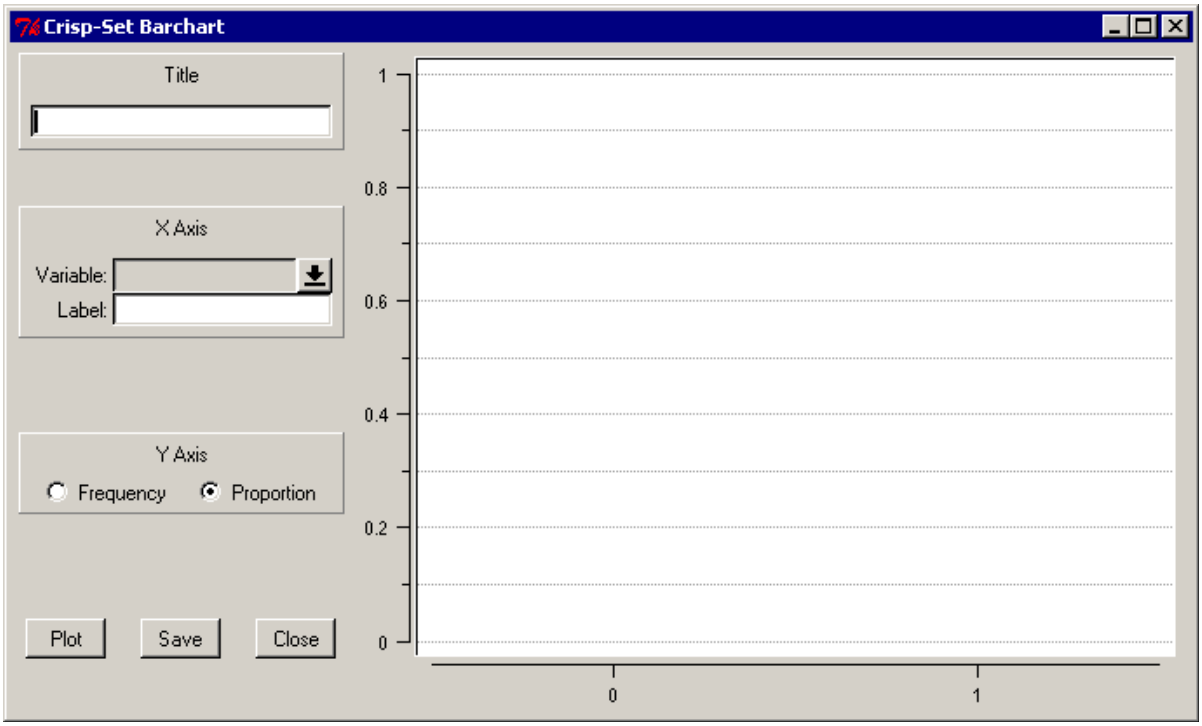
D) グラフ

棒グラフ

Barchart（棒グラフ）という機能では、事例のグループを要約する単純な棒グラフが表示される。

- **棒グラフ**を作成するには、次のように選択する:
  Graphs（グラフ）
    Crisp（クリスプ集合）
      Barchart（棒グラフ）

- 次のようなウィンドウが開く。



- グラフに表示される X 軸のカテゴリーを定義するために、変数を選択する。選択することができるのは**数値のデータ**のみである。変数のそれぞれの値につき、棒グラフの棒が 1 つ作

成される。

- この棒グラフ機能は、**0 から 2 の間の値**しか表示しない。値が整数でない場合、棒は重なり合う。ドント・ケア（don't care）とコーディングされている変数を持つ事例は、2 の値を持っているものとして表示される。

- オプションとして、変数の **Label（ラベル）**やグラフの **Title（タイトル）**を入力して、グラフに表示させることができる。

- そのカテゴリーの **Frequency（事例の度数）**と **Proportion（事例の割合）**のどちらを Y 軸に表示するかを決定する。

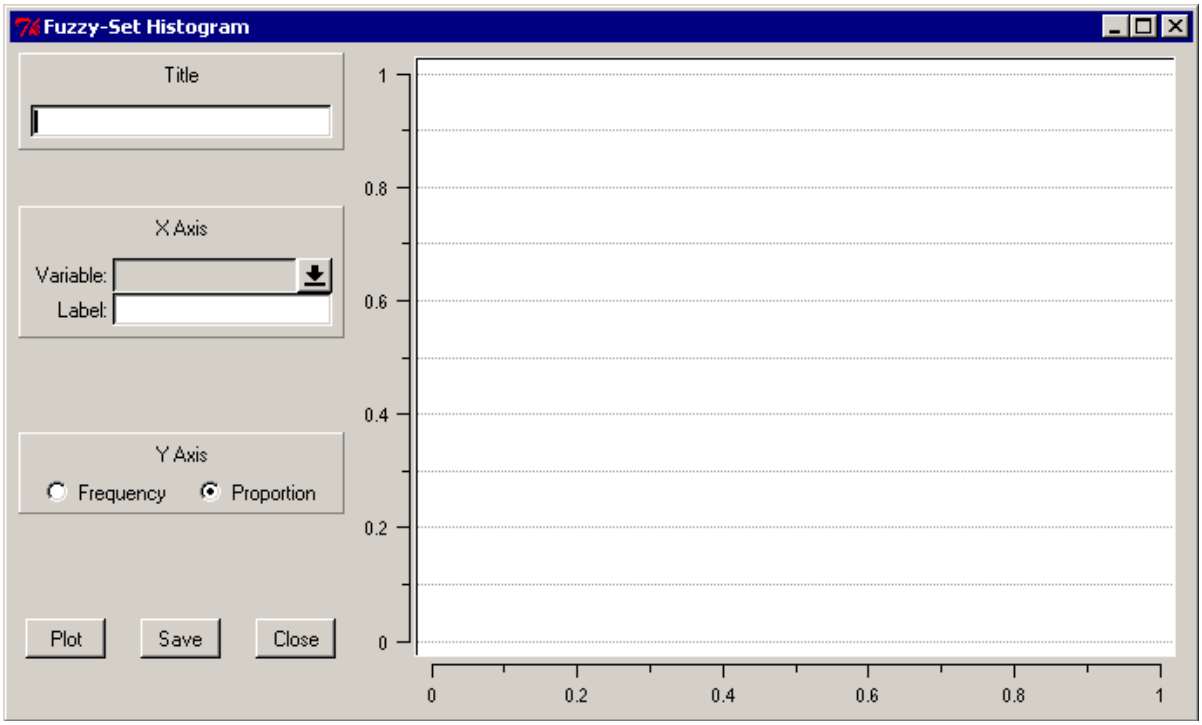
- 指定し終わったら、**Plot（プロット）**ボタンをクリックする。右の白いフィールドにグラフが表示される。

- このグラフを**印刷する**には、ポストスクリプト（PostScript）ファイル（*.ps）として**保存する**必要がある。Save（保存）ボタンをクリックし、保存オプションから「Postscript file」を選択する。Ghostview（GSview）などポストスクリプトファイルを読めるプログラムでそれを開く（Ghostviewはインターネットで無料でダウンロードできる。http://www.cs.wisc.edu/~ghost/）。

ヒストグラム

- ヒストグラムを作成するには、次のように選択する:
  Graphs（グラフ）
    Fuzzy（ファジィ集合）
      Histogram（ヒストグラム）

- 次のようなウィンドウが開く。



- グラフに表示される X 軸のカテゴリーを定義するために、変数を選択する。選択することができるのは**数値のデータ**のみである。変数のそれぞれの値につき、ヒストグラムの棒が 1 つ作成される。

- オプションとして、変数の **Label（ラベル）**やグラフの **Title（タイトル）**を入力して、グラフに表示させることができる。

- そのカテゴリーの **Frequency（事例の度数）**と **Proportion（事例の割合）**のどちらを Y 軸に表示化を決定する。

- 指定し終わったら、**Plot（プロット）**ボタンをクリックする。右の白いフィールドにグラフが表示される。

- このグラフを**印刷する**には、ポストスクリプト（PostScript）ファイル（*.ps）として**保存する**必要がある。Save（保存）ボタンをクリックし、保存オプションから「Postscript file」を選択する。Ghostview（GSview）やいくつかのバージョンのWordなど、ポストスクリプトファイルを読めるプログラムでそれを開く（Ghostviewはインターネットで無料でダウンロードできる。http://www.cs.wisc.edu/~ghost/）。

XY プロット（散布図）

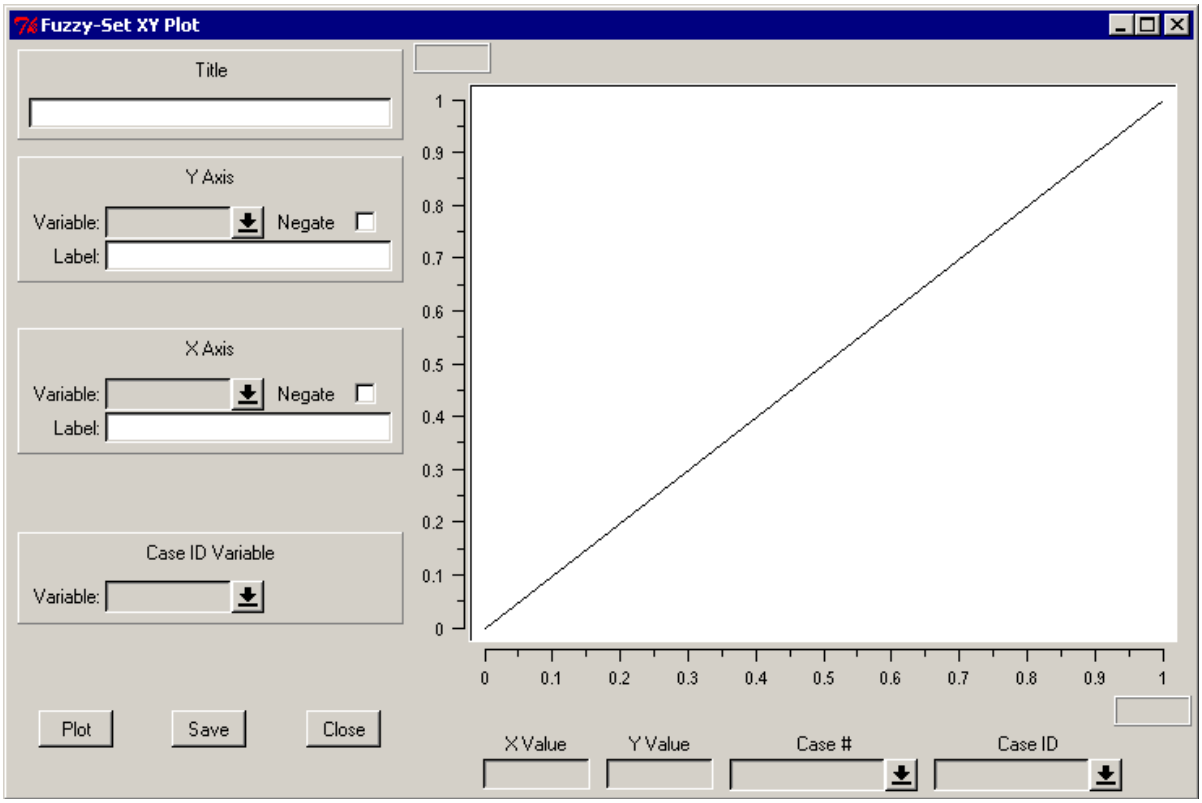
- **XY プロット（散布図）**を作成するには、次のように選択する:

Graphs（グラフ）

Fuzzy（ファジィ集合）

XY Plot（XY プロット）

➢ 次のようなウィンドウが開く。



➢ グラフに表示される **X Axis（X 軸）**の値を定義するために、変数を選択する。0 以上 1 以下の数値をとる変数（すなわちファジィ集合）のみを使用することができる。

➢ グラフに表示される **Y Axix（Y 軸）**の値を定義するために、変数を選択する。同様に、0 以上 1 以下の数値をとる変数のみを使用することができる。

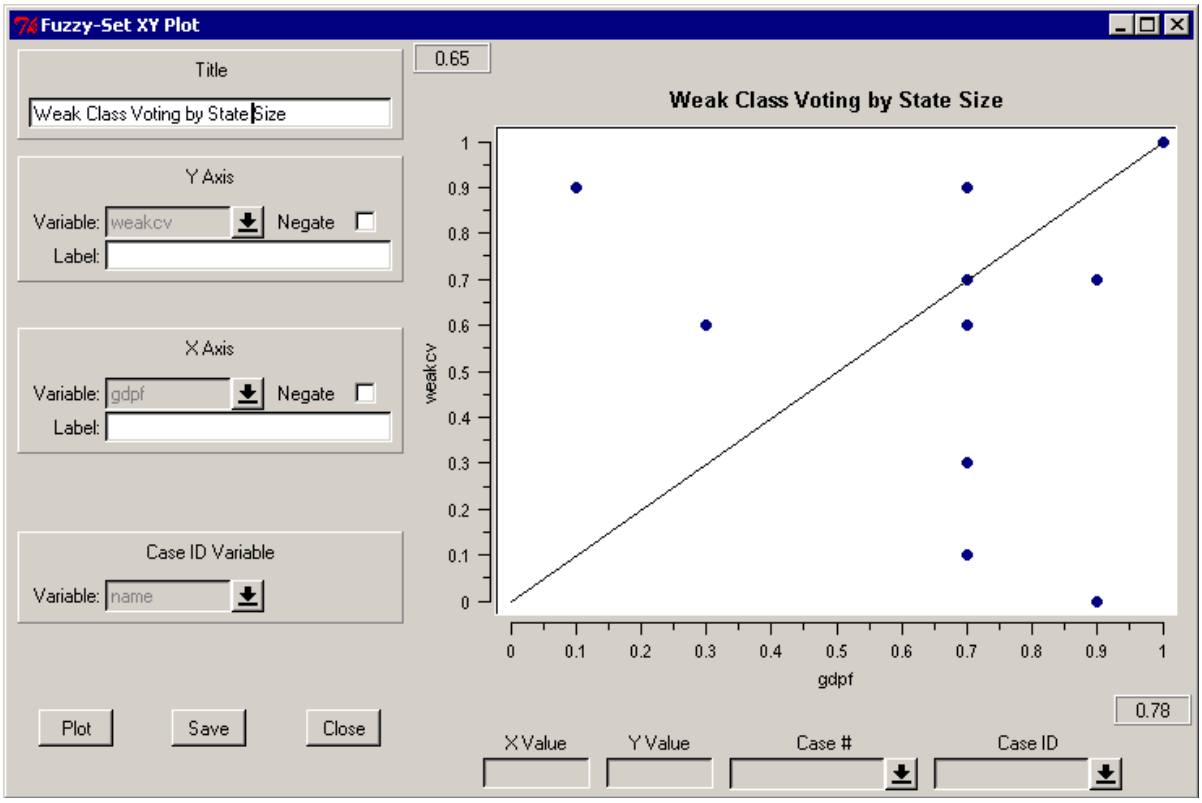
➢ オプションとして、グラフの **Title（タイトル）**を入力して、保存したときにグラフに表示させることができる。

➢ **Case ID Variable（事例 ID 変数）**を選択して、さらなる情報を付け加えることもできる。この変数はグラフには表示されないが、グラフの特定の点をクリックすることでその変数の値を決定することができる。事例 ID は、右下の隅にある Case ID（事例 ID）と書かれたフィールドに表示される。事例 ID 変数の例としては、データセット中の国の名前という文字列変数が挙げられる。グラフを一旦プロットすれば、グラフ中のどの点もクリックすること

ができるようになる。グラフの下の Case ID（事例 ID）のフィールドに、この点の表す国名などが表示される。そのフィールドの隣の矢印をクリックすると、グラフ中に同じ値を持つ複数の国がある場合にそれを表示することができる。

- グラフの下にある欄では、特定の事例の x と y の値や、それぞれの点に対応する事例の数も表示される。

- 指定し終わったら、**Plot**（プロット）ボタンをクリックすれば次のようなグラフが表示される。



- グラフの X 軸と Y 軸の隅にある四角の中の数字は、集合論的な整合度（consistency score）を表している。右下の四角の方の数字は、Y≤X（Y は X の部分集合）と整合的なデータの度合いを表している。左上の四角の方の数字は、X≤Y（X は Y の部分集合）と整合的なデータの度合いを表している。これらの 2 つの数字のうち 1 つが高い整合度を示す場合、もう 1 つの方の数字は被覆度（coverage score）と解釈できる。例えば、左上の数字が 0.91 で右下の数字が 0.63 の場合、データは X が Y の部分集合であるという議論と非常に整合的である。そして Y の被覆度は 63%、すなわち Y のメンバーシップ度の合計の 63%を X が説明している。

- Variable（変数）の隣にある Negate（否定）というオプションをクリックすると、その変

数の否定をグラフで使うことができる。この機能は、1 から当該変数を引くという計算を行っている（例: Pressure=0.33 なら、その否定は 0.67。[~ や fuzzynot(x)と同じである。]）

- このグラフを**印刷する**には、ポストスクリプト（PostScript）ファイル（*.ps）として**保存する**必要がある。Save（保存）ボタンをクリックし、保存オプションから「Postscript file」を選択する。Ghostview（GSview）などポストスクリプトファイルを読めるプログラムでそれを開く（Ghostviewはインターネットで無料でダウンロードできる。http://www.cs.wisc.edu/~ghost/）

4. クリスプ集合分析

本パートでは、通常の集合（クリスプ集合）への事例の帰属関係を表す2値の社会データの分析について述べる。この手法の詳しい議論については、Ragin (1987)やRagin (2000)の第5章を参照してほしい。ここで用いられるデータ分析の戦略は、質的比較分析（qualitative comparative analysis, QCAと略す）として知られている。QCAはブール代数の計算を基礎としている。ブール代数においては、それぞれの事例は集合に属するか属さないかのどちらかである。したがってQCAでは、1が当該集合への帰属（membership）、0が非帰属（nonmembership）を表すような2値データを使用する。通常の集合（クリスプ集合）を用いるQCAは、csQCAとも呼ばれる。

A) 基本概念

ブール代数には、質的比較のための明確な代数的基礎が存在する。ブール代数は論理代数や集合代数とも呼ばれ、19世紀中頃にGeorge Booleによって開発された。質的比較分析で用いられるブール代数の法則は、きわめて簡単なものである。ブール代数の以下の7つの側面がアルゴリズムにとって不可欠である。ここでは、簡単な概念からより難解な概念へという大まかな順序でそれらを提示する。

1) 2値データの使用

ブール代数には、真（または「あり」）と偽（または「なし」）という2つの状態がある。これらの2つの状態は、2を基数として表される。すなわち、「あり」の場合は1、「なし」の場合は0という具合である。ブール代数を基礎とする典型的な比較分析では、ある結果が得られたとき（すなわち、ある結果が真のとき）に、どのような条件が存在していたのか、あるいは存在していなかったのかということを扱う。したがって、社会データのブール代数分析では、独立変数も従属変数も全て名義尺度でなければならない。間隔尺度は、複数のカテゴリーを持つ名義尺度に変換される。2つ以上のカテゴリーを持つ名義尺度は、いくつかの2値変数で表される。

2) ブール否定

ブール代数において否定（negation）を計算するには、ブール値（メンバーシップ度）を0のものは1へ、1のものは0へと切り替える。例えば、「男性」のクリスプ集合の否定は、「男性でない」のクリスプ集合である。もしある事例の「男性」のクリスプ集合についてのブール値が1であるならば、「男性でない」のクリスプ集合についてのブール値は0になる。

3) データを表現するための真理表の使用

質的比較のテクニックとしてブール代数を使うためには、素データ（raw data）を真理表（truth table）として再構成する必要がある。真理表の背後にあるアイディアは単純なものである。データを名義尺度変数に再コーディングして2値（1と0）の形で表せば、あとはそのデータを、独立変数の値の組み合わせごとにソートするだけでよい。独立変数の値の論理的な組み合わせの1つ1つは、真理表の1行として表される。真理表のこの部分ができあがったら、それぞれの行に出力値（0か1の値の従属変数）を割り振る。この出力値は、入力値の組み合わせ（独立変数の値の組み合わせ）が共通する事例の、結果の値を基礎とするものである。このようにして、入力値（独立変数）の様々な組み合わせとそれに対応する出力値（従属変数）が、真理表で要約される。

真理表の行数は、原因条件の値についての論理的に可能な組み合わせの数と同じである。例えば、3つの2値変数がある場合、3つの独立変数の「あり／なし」についての論理的に可能な組み合わせを考えれば、真理表は$2^3 = 8$行となる。3つの2値独立変数と1つの2値従属変数（1＝「あり」, 0＝「なし」）を持つ、中程度の規模のデータセットの真理表は、表1のようになる。厳密に言うと、それぞれの組み合わせの事例数を真理表に書く必要はない。この例で事例数を記したのは、それぞれの行は単一の事例ではなく、特定の入力値の組み合わせを持つ全ての事例の要約であるということを読者に想起させるためである。この点では、真理表の行は、いくつかのカテゴリカル独立変数についての多重クロス表のセルのようなものだと言える。

表1: 軍事政権の崩壊の3つの原因を表す仮想的な真理表

条件			軍事政権の崩壊	事例数
conflict	death	cia	failure	
0	0	0	0	9
1	0	0	1	2
0	1	0	1	3
0	0	1	1	1
1	1	0	1	2
1	0	1	1	1

0	1	1	1	1
1	1	1	1	3

conflict = 年配の将校と若年の将校の対立
death = 強権的な独裁者の死
cia = 体制に対するCIAの不満

4) グループ化

論理的に可能な組み合わせの数を計算できる(2^k)のとちょうど同じように、論理的に可能なグループの数も計算できる。公式は$3^k - 1$である（kは原因条件の数を表す)。表2は、表1で見た3つの2値変数についての、論理的に可能な26個のグループを表している($3^3 - 1 = 26$)。今説明した公式を使うと、次のような26個の可能なグループがあることが分かる。すなわち、3つの原因条件の組み合わせによる8個のグループ、2つの原因条件の組み合わせによる12個のグループ、1つの原因条件による6個のグループ、という26個である

表2: 3つの2値変数を使ったグループ（表1をもとに）

初期的な条件組み合わせ(3つの原因条件による8個の組み合わせ)	2つの原因条件を組み合わせたグループ(12個)	1つの原因条件のグループ(6個)
conflict*death*cia	conflict*death	conflict
conflict*death*~cia	conflict*~death	~conflict
conflict*~death*cia	~conflict*death	death
conflict*~death*~cia	~conflict*~death	~death
~conflict*death*cia	conflict*cia	cia
~conflict*death*~cia	conflict*~cia	~cia
~conflict*~death*cia	~conflict*cia	
~conflict*~death*~cia	~conflict*~cia	
	death*cia	
	death*~cia	
	~death*cia	
	~death*~cia	

5) ブール和

ブール代数では、A + B = ZでA = 1かつB = 1ならば、Z = 1である。すなわち、1 + 1 = 1となる。ブール和の基本的なアイディアは、足し算する項のいずれか1つでも満たされれば（存在すれば)、結果が真となる（生じる）というものである。ブール代数の和は、論理演算子の「または」（OR）に相当する（ここでの議論では、大文字のORを論理演算子のORを示すものとして使

っている)。今までの説明より、A + B = Zについて次のことが言える。A = 1またはB = 1ならば、Z = 1である。

この法則を理解する最良の方法は、算術的に考えるのではなく、論理学の用語で考えてみることである。例えば、仕事を首になるような行動には様々なものがあるだろう。そのうちのいくつを行ったかは問題ではない。そのうちのどれか1つでも行えば、首になる。2つ行ったからといって1つだけ行った場合よりも首になりやすくなるわけではない。どちらでも首は首であり、純粋に質的な状態だからである。この例は、ブール和の性質を簡潔に説明している。すなわち、足し算する条件のどれか1つでも満たされれば、予想される結果が生じる、という性質である。ブール和のこの側面は、社会科学の分析、特に質的比較においてとても有用なのであるが、その価値は一般にはあまり認識されていない。

軍事政権の崩壊を考えてみよう。軍事政権の崩壊を引き起こす次のような3つの一般的原因があると仮定する。年配将校と若年将校の鋭い対立（conflict)、強権的な独裁者の死（death)、体制に対するCIAの不満（cia)、という3つである。これらの3つの原因のうちどれか1つでも満たされれば、政権が崩壊するのに十分かもしれない。表1には、様々な国のそういった多数の政権についての真理表が示されている（1 =「あり」で0 =「なし」である)。原因条件のそれぞれの組み合わせは、政権の崩壊を産み出すものと、産み出さないものがある。この真理表には矛盾する行はない。
「簡単化された」ブール式は、次のようになる。

failure = conflict + death + cia

この式は、政権が崩壊した場合としなかった場合の両方について、3つの原因条件と政権の崩壊の関係を、単純でエレガントに表現している。次のように単純に言える。これらの原因条件のうちどれか1つ（あるいは2つ、3つ）が満たされれば、軍事政権は崩壊する。

6) ブール積

ブール積は、通常の積とはかなり異なる。ブール代数を社会科学へ適用する場合、通常、「積和」（sums of products）と呼ばれる式を簡単化していく必要があるので、ブール積には実際的な重要性がある。ここで積（product）とは、原因条件の特定の組み合わせのことである。表1の軍事政権の崩壊についてのデータは、「原始的な」（primitive）（すなわち、縮約していない）積和形で、次のように表現することができる。

```
failure = conflict*~death*~cia +
          ~conflict*death*~cia +
          ~conflict*~death*cia +
```

conflict*death*~cia +
conflict*~death*cia +
~conflict*death*cia +
conflict*death*cia

7つの項はそれぞれ、軍事政権崩壊の事例が少なくとも1つは観察されたような原因条件の組み合わせを表している。それぞれの項は、条件の交わり合い（intersection）（原因条件の存在と原因条件の欠如の結合）を表しているので、積の形で表現されている。すなわち上の式は、軍事政権の崩壊につながるような、条件の原始的な組み合わせを表している。

ブール積もブール和と同じように、算術的なものではない。conflict*~death*~ciaという式は、conflict(1)とdeath(0)とcia(0)という値を掛け算して0という値になるという意味ではない。この式は、単にconflictの存在とdeathの欠如とciaの欠如を組み合わせるという意味である。それらを合わせたfailure= conflict*~death*~ciaという状況は、データにおいて2回起こっている。ブール積のこの結合的な性質から、上の原始的な積和形の式は次のように解釈できる。すなわち、3つの原因の7つの組み合わせのうちのどれか1つでも満たされれば、failure（軍事政権の崩壊）が生じる。ブール代数においては、和は論理演算子の「または」（OR）を意味し、積は論理演算子の「かつ」（AND）を意味する。つまり、3つの原因が様々な仕方で「かつ」で結合され、様々な経験的な条件の組み合わせが産み出される。この条件の交わり合いは、「または」で結び付けられ、縮約されていない積和形の式が形成される。そうしてできあがった式は、軍事政権の崩壊につながる3つの原因の様々な組み合わせを表現しているのである。

7）組み合わせ論理

ブール代数分析では、仕様上、組み合わせ論理（combinatorial logic）を用いることになる。上述の軍事政権の崩壊の分析では、真理表（表1）の最初の4行の原因条件のみを調べて、3つの原因のうちの1つでも存在すれば軍事政権は崩壊する、と結論できるように一見思われるかもしれない。このようなショートカットをしたくなるところだが、ブール代数分析はデータをはるかに厳密に利用する。なぜなら、ブール代数分析では、原因の欠如は、原因の存在と同様の論理的地位にあるからである。上述のように、ブール積とは、原因の存在や欠如を組み合わせた、交わり合い（intersection）を指す。

真理表（表1）の2番目の行を考えてみよう。この行は、軍事政権崩壊の2事例が、conflict*~death*~ciaという原因条件の組み合わせと関係していることを示している。少し調べれば、failure（軍事政権崩壊）は、1番目の原因条件であるconflictによって生じたとわかる。しかし、注意しなければならないのは、もし研究者が真理表のこの行以外の軍事政権崩壊の事例の情報を全く持っていないならば、conflictがfailureを引き起こすのはdeathとciaが欠如している場合のみであると結論するかもしれない、ということである。これが、conflict*~death*~ciaが意味していることである。この行自体からは、deathやciaの一方または両方が存在する場合に

conflictがfailureを引き起こすかどうかは分からない。conflict*~death*~ciaの2事例から研究者が分かることは、conflictがfailureを引き起こすには、他の原因条件（deathとcia）が欠如していることが必要であるかもしれない、ということだけである。ブール代数の観点からは、これら他の条件のうち一方または両方が存在する場合（例えば、conflict*~death*ciaという条件の組み合わせ）に、failureが生じないかもしれないということは、十分考えられることである。条件の内容に戻って考えてみれば、CIAの干渉が存在する場合（cia）、そういった部外者の干渉の試みに反対して若年将校と年配将校が結束し、両者の対立（conflict）が解消するかもしれないのである。

この議論をさらに推し進めるために、研究者が真理表の最初の4行の知識のみを有していたと仮定してみよう。データからは、3つの原因条件のうちのどれか1つでも存在すればfailureが生じるという見解が支持されうるが、同様に、deathやciaが欠如しているときのみconflictからfailureが生じる(conflict*~death*~cia）、conflictやciaが欠如しているときのみdeathからfailureが生じる(~conflict*death*~cia)などといった可能性もある。組み合わせ論理を厳密に適用すると、限られた種類の事例から引き出した結論にはこういった制限が課せられる。

組み合わせ論理のこの特徴は、事例（特にその中でも原因条件の関連する特徴）を全体論的に見るべきであるという考え方と整合的である。ブール代数アプローチの全体論的な性質は、コンテクストの中で様々な原因条件を調査しているような、社会科学の比較研究において質的分析を行う研究者たちの指向とも一致する。真理表（表1）の2番目の行を調べるとき、conflictがfailureを引き起こすと解釈するのではなく、conflict*~death*~ciaがfailureを引き起こすと解釈する。このように、ブール代数を基礎とした質的比較においては、原因条件は個々に分離して見るのではなく、他の関連する原因条件が存在しているか欠如しているかというコンテクストの中で常に見なければならない。

簡単化

組み合わせ論理の制限的な性質から、ブール代数アプローチは複雑さの上に複雑さを重ねているだけなのではないかと思われるかもしれないが、それは違う。複雑さを簡単化するような単純明快な規則が存在する。それは、原始的なブール項を縮約して、より簡潔な表現にするような規則である。そのような規則のうちで最も基本となるのは次のようなものである。すなわち、2つのブール項[1]（Boolean expression）において1つの原因条件だけが異なっており結果は同じとなるならば、その2つの項において異なっている原因条件は無関係であり、それを取り除くことでより単純な、結合された項にすることができる。

[1]（訳注）ここでは、積和形のブール式の積項のことを指している。expression は「式」と日本語訳されることが多いが、日本語の「式」とは表現するものが正確には異なる。すなわち、expression は x や x-5 のようなものを指し、x=8 や x-5=3 のようなイコールを含む式は equation と呼んで区別する。そこで、このマニュアルでは 2 つを区別するために、Boolean expression を「ブール項」と訳し、Boolean equation を「ブール式」と訳すことにした。

要するに、この簡単化（minimization）の規則によって、1つの原因条件だけが異なっているような2つのブール項から、1つの結合されたブール項を生み出すことができる。例えば、conflict*~death*~ciaとconflict*death*~ciaは、両方ともfailureという結果を引き起こし、deathという条件のみが異なっている。そして、他の要素は全て同じである。上述の簡単化の規則から、この2つの項は、conflict*~ciaという単一のより簡潔な項に置き換えることができる。言い換えれば、conflict*~death*~ciaとconflict*death*~ciaという真理表の2行を比較することで、conflict*~ciaの場合においては、deathの値は無関係であることが分かる。deathという原因条件が存在していても欠如していても、failureは起こる。

この単純なデータ縮約の論理は、実験計画の論理とパラレルになっている。deathという原因条件のみが異なり、結果には何の違いもない（なぜなら、conflict*~death*~ciaもconflict*death*~ciaもfailureが起こった場合だからである）。実験計画の論理によれば、conflict*~ciaが存在する場合（すなわち、これら2つの条件を不変とする場合）は、deathはfailureとは無関係である。このように、ブール代数の簡単化のプロセスは、実験計画の論理によく似たものになっている。これは、理想的な社会科学の比較研究の論理を、ストレートに操作化したものであると言える。

ブール代数の簡単化のプロセスは、これ以上ブール項を縮約できないところまで、複雑な項から簡潔な項へとボトムアップ式に、段階をおって行われる。上で挙げた、軍事政権崩壊のデータをもう一度考えてみよう。1つの原因条件が存在しそれ以外の2つが欠如した行は、2つの原因条件が存在しあとの1つが欠如した行と結合できる。なぜなら、これらの行は同じ結果（failure）となり、それぞれのペアは1つの原因条件のみが異なっているからである。したがって、次のように結合できる。

conflict*~death*~ciaと、conflict*death*~ciaが結合して、conflict*~cia
conflict*~death*~ciaと、conflict*~death*ciaが結合して、conflict*~death
~conflict*death*~ciaと、conflict*death*~ciaが結合して、death*~cia
~conflict*death*~ciaと、~conflict*death*ciaが結合して、~conflict*death
~conflict*~death*ciaと、conflict*~death*ciaが結合して、~death*cia
~conflict*~death*ciaと、~conflict*death*ciaが結合して、~conflict*cia

同様に、2つの原因条件が存在してあとの1つが欠如した行は、3つが全て存在する行と次のように結合できる。

conflict*death*~ciaと、conflict*death*ciaが結合して、conflict*death
conflict*~death*ciaと、conflict*death*ciaが結合して、conflict*cia
~conflict*death*ciaと、conflict*death*ciaが結合して、death*cia

さらなる縮約が可能である。最初に縮約によって求められた6つの項は、次に求められた3つの項と下のように結合して、さらに簡単なブール項が生み出されることに注目してほしい。

conflict:~deathと、conflict*deathが結合して、conflict
conflict*~ciaと、conflict*ciaが結合して、conflict
~conflict*deathと、conflict*deathが結合して、death
death*~ciaと、death*ciaが結合して、death
~conflict*ciaと、conflict*ciaが結合して、cia
~death*ciaと、death*ciaが結合して、cia

このように冗長で退屈ではあるが、方法自体は単純な簡単化のプロセスによって、次のような最終的なブール式（Boolean equation）が導かれる。

failure = conflict + death + cia

もちろん、この式は単に真理表全体を調べるだけでも明白だったのだが、この例を選んだのは、単純だからである。この例ならば、ブール代数の簡単化についての鍵となる特徴がストレートに分かる。簡単化はボトムアップ（つまり、帰納的な指向を持つ方式）なのである。そこでは、結果が真となる条件のより広い集合（すなわち、原因条件のより単純な組み合わせ）を見つけようとする。そして、それはまるで実験のように、1つの原因条件だけが異なるような条件の組み合わせのペアに焦点を当てるものである。

1)「主項」の使用

さらに、論理包含（含意、implication）というブール代数の概念を導入する。一方のブール項が他方のブール項の部分集合になっている場合、後者の項は前者の項を論理包含しているという。例えば、aはa*~b*~cのメンバー全てを含む（すなわち、a*~b*~cはaの部分集合である）ので、aはa*~b*~cを論理包含する。この概念は、次のような例で理解するのがよいだろう。aが経済的な従属国、bが重工業の存在、cが中央調整型の経済を表すならば、aは全ての経済的従属国を指すのに対し、a*~b*~cは重工業も中央調整型経済も欠いた経済的従属国を指すことになる。明らかに、a*~b*~cのメンバーはaに含まれる。このように、aはa*~b*~cを論理包含する。

論理包含の概念は明白なものに見えるが、原始的な積和形の式を簡単化するための重要な道具となる。表3のような仮想的な真理表を考えてみよう。この表は、発生したストライキの成功（success）に影響を与えると思われる、次のような3つの原因条件をまとめたものである。すなわち、ストライキをしている人々が生産した製品の市場の好景気（market）、関連した産業の労働者による支援ストの脅威（threat）、多額のスト資金の存在（fund）という3つの条件である。

success（ストライキの成功）についての、縮約されていない（原始的な）ブール項を使ったブール式は、次のようになる。

success = market*~threat*fund + ~market*threat*~fund +
market*threat*~fund + market*threat*fund

表3: ストライキが成功する3つの原因条件を表した仮想的な真理表

条件			結果	度数
market	threat	fund	success	
1	0	1	1	6
0	1	0	1	5
1	1	0	1	2
1	1	1	1	3
1	0	0	0	9
0	0	1	0	6
0	1	1	0	3
0	0	0	0	4

これらのデータをブール代数で分析する際の最初のステップは、真理表の行をできるだけ多く結合しようとすることである（簡単化のプロセスのこのパートでは、出力値が1、すなわちストライキが成功した行を使用することに注意)。真理表の簡単化のこの最初のステップで、3変数の項を4つ持つ原始的なブール式が、2変数の項を3つ持つ式になる。すなわち、次のような部分的に簡単化されたブール式ができあがる。

market*threat*fundと、market*~threat*fundが結合して、market*fund
market*threat*fundと、market*threat*~fundが結合して、market*threat
market*threat*~fundと、~market*threat*~fundが結合して、threat*~fund

success = market*fund + market*threat + threat*~fund

上の式のように、単純な簡単化の規則（同じ出力値を持ち原因条件が1つだけ異なった行を結合する）を使って作った積項を、主項（素数的条件、prime implicant）と呼ぶ。通常は、それぞれの主項は、真理表のいくつかの原始的なブール項をカバー（すなわち、論理包含）する。例えば、上の部分的に簡単化された式では、market*fundは、market*threat*fundとmarket*~threat*fundという、真理表にリストアップされた2つの原始的なブール項をカバーしている。

この部分的に縮約されたブール項は、次のようなブール代数分析の一般的な知見の例となっている。すなわち、もとの原始的なブール項全てをカバーするのに必要なよりも、たくさんの縮約されたブール項（主項）が使われていることが多い。例えば、market*threatという主項はmarket*threat*fundとmarket*~threat*fundという原始的な項を論理包含するが、これら2つの項は、market*fundとthreat*~fundでもそれぞれカバーされている。したがって、market*threatは、純論理的な観点からは冗長であるかもしれない。すなわち、この項は、必要不可欠な主項でない可能性がある。どの主項が論理的に必要不可欠であるかを決定するには、主項表（prime implicant chart）という簡単化の装置を使う。主項表による簡単化は、ブール代数の簡単化の第2段階である。この第2段階を行うかどうかは自由選択である。

要約して言えば、簡単化のプロセスのこの第2段階の目標は、論理的に最小の数の主項で、できる限り多くの原始的なブール項を「カバー」することである。これは、冗長さを避けたいというストレートな願望から来るものである。主項表は、主項と原始的なブール項のつながりを図示する。ストライキの結果についてのデータでこのつながりを図示した主項表が、表4である。少し調べれば、もとの原始的なブール項を全てカバーするのに必要な主項の最小数は、2であることが分かる（非常に複雑な主項表の場合には、洗練されたコンピューターアルゴリズムが必要になる。Mendelson (1970), Roth (1975), McDermott (1985)を参照。）market*fundとthreat*~fundという主項は、4つの原始的なブール項を全てカバーしている。したがって、主項表の分析により、論理的に不可欠な主項のみを含むような、最終的な縮約されたブール項は次のようになる。

success = market*fund + threat*~fund

この式は、ストライキをしている人々が生産した製品の市場が好景気で、かつ、多額のスト資金がある場合（market*fund）か、または、関連した産業の労働者による支援ストの脅威があり、かつ、スト資金が少ない場合（threat*~fund）に、ストライキが成功するということを簡潔に表現している（恐らく、ストライキをしている労働者が他の労働者の支援を非常に必要にしている場合にのみ、支援ストの脅威が深刻に捉えられるのであろう。）

表4: 主項がカバーしているもとの項を示す主項表（ストライキについての仮想データ）

		原始的なブール項			
		market* threat* fund	market* ~threat* fund	market* threat* ~fund	~market* threat* ~fund
主項	market*fund	○	○		
	market*threat	○		○	
	threat*~fund			○	○

これらの単純な手続きにより、研究者は、結果と関連する原因条件の様々な組み合わせを表す

ような、論理的に最も簡単な式を見つけることができる。最終的な縮約された式では、2つの（論理的に最も簡単な）ストライキを成功させる原因条件の組み合わせが表されており、多元結合因果[2]（multiple conjunctural causation）について明確に述べられている。

2）ド・モルガンの法則の利用

ド・モルガンの法則の適用の仕方は簡単である。上述した、ストライキの成功についての仮想的な分析における解について考えてみよう。解は、success = market*fund + threat*~fundであった。縮約された式において「あり」とされていた要素（例えば、market*fundという項のmarket）は「なし」と書き換えられ、「なし」とされていた要素（例えば、threat*~fundという項の~fund）は「あり」と書き換えられる。次に、論理積（「かつ」）は論理和（「または」）に書き換えられ、論理和は論理積に書き換えられる。これら2つの規則を適用すると、

success = market*fund + threat*~fund

は次のように書き換えられる。

~success = (~market + ~fund)*(~threat + fund)

= ~market*~threat + ~market*fund + ~threat*~fund

この式によれば、(1) ストライキをしている人々が生産した製品の市場が好景気でなく、かつ、支援ストの脅威がそれほどない、または (2) ストライキをしている人々が生産した製品の市場が好景気でなく、かつ多額のスト資金がある、または (3) 支援ストの脅威がそれほどなく、かつ、少額のスト資金しかないという場合に、ストライキは失敗する（~market*fundという組み合わせ―好景気でない市場と多額のスト資金―は、矛盾しているように見えるが、経済が安定期の後に下降した場合を示しているのかもしれない。この状況では、休業は経営者に歓迎されてしまう可能性がある。）

ド・モルガンの法則は、与えられた論理式の厳密な否定を導く。真理表に「残余部分」（remainder）の組み合わせがあり、それらを「ドント・ケア」（don't care）として使用する場合、ド・モルガンの法則を適用して得られる論理式は、結果が「なし」の場合を分析して得られる式と同じにはならない。同様に、最初の分析で残余部分を「偽」（false）として定義した場合、（ポジティブな事例についての）解にド・モルガンの法則を適用すると、ネガティブな事例だけ

[2]（訳注）多元結合因果（multiple conjunctural causation）は、多元因果（multiple causation）と結合因果（conjunctural causation）を組み合わせた言葉である。多元因果は複数の原因条件が同じ結果をもたらすような関係をいう。すなわち、論理式の「または」にあたる関係である。例えば、原因条件 A または B が存在すれば結果 S がもたらされる場合が、多元因果である。結合因果は、複数の原因条件が同時に存在した場合にのみ、結果がもたらされるような関係をいう。すなわち、論理式の「かつ」にあたる関係である。例えば、原因条件 C かつ D が存在すれば結果 S がもたらされる場合が、結合因果である。多元結合因果は以上を組み合わせた、例えば、原因条件「A かつ B」、または「C かつ D」が存在する場合に、結果がもたらされる、といった関係をいう。

でなく残余部分も含んだ論理式が得られる。

3）必要条件と十分条件

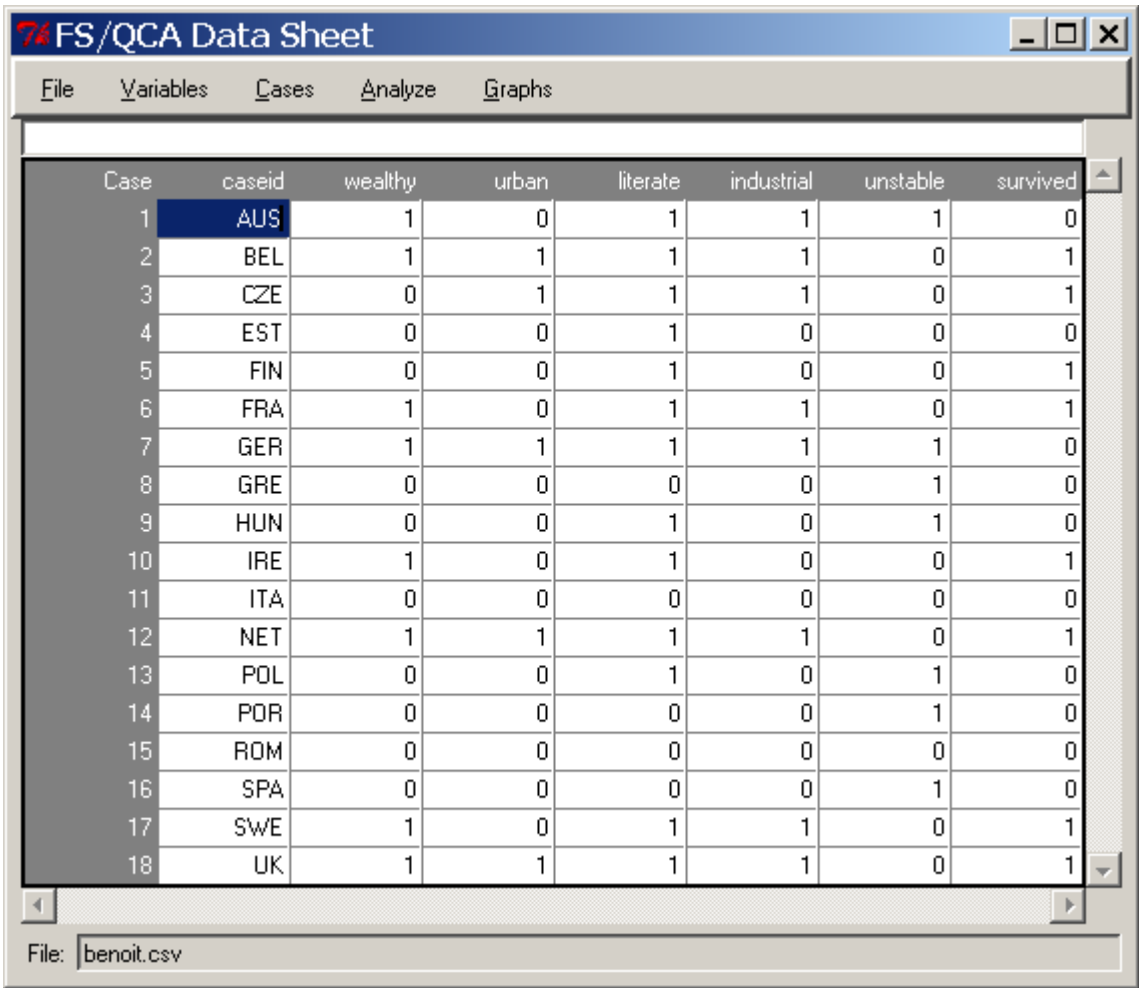
ある結果が生じる場合に当該原因条件が必ず存在するならば、その原因条件は必要条件であると定義される。当該原因条件がある結果を生じうるならば、その原因条件は十分条件であると定義される。この区別は、理論的観点と関連させた場合にのみ意味がある。例えば、どのような原因条件も，それを原因条件として特定する理論がなければ、必要条件とはなりえない。必要性も十分性も、原因条件を提示する理論から独立して存在することはない。

必要性も十分性も、通常は一緒に考えられる。なぜなら、必要性と十分性という2つの組み合わせの全てに意味があるからである。ある原因条件が，結果を生じる唯一のものであり、単一である（すなわち、原因の組み合わせではない）ならば、その原因条件は必要十分条件である。ある原因条件が、結果を生じうるがこの可能性を持つ唯一の原因条件ではない場合、その原因条件は十分条件ではあるが必要条件ではない。ある原因条件が、他の原因と組み合わさって結果を生じさせ、そのような組み合わせ全てに現れるならば、その原因条件は必要条件ではあるが十分条件ではない。最後に、ある原因条件が、結果を生み出す条件の組み合わせの部分集合の中のみに現れるならば、その原因条件は必要条件でも十分条件でもない。以上のように、全部で4つのカテゴリーの原因がある（十分性のあり・なしと必要性のあり・なしのクロス表が形成される。）

通常、QCA（クリスプ集合またはファジィー集合）の適用により、結果が生じる十分条件の組み合わせに関する論理式が得られる。リスト化された組み合わせは網羅的かもしれないし、網羅的でないかもしれない。すなわち、結果を生じる事例の全てを説明するものではないかもしれない。通常、十分条件の組み合わせの分析の前に、必要条件を別個に検討するのが最良である。もし必要条件が見つかり、それが必要条件として道理にかなったものであるならば、真理表の分析（十分条件の組み合わせの分析）は省略することができる。しかし、そのような条件も結果の提示の際に依然として重要な役割を果たし、真理表の分析において十分条件とされた原因条件の組み合わせの構成要素と見なされる可能性がある。

B）データ

以下のウィンドウは、クリスプ集合のデータシートのサンプルである。



Case	caseid	wealthy	urban	literate	industrial	unstable	survived
1	AUS	1	0	1	1	1	0
2	BEL	1	1	1	1	0	1
3	CZE	0	1	1	1	0	1
4	EST	0	0	1	0	0	0
5	FIN	0	0	1	0	0	1
6	FRA	1	0	1	1	0	1
7	GER	1	1	1	1	1	0
8	GRE	0	0	0	0	1	0
9	HUN	0	0	1	0	1	0
10	IRE	1	0	1	0	0	1
11	ITA	0	0	0	0	0	0
12	NET	1	1	1	1	0	1
13	POL	0	0	1	0	1	0
14	POR	0	0	0	0	1	0
15	ROM	0	0	0	0	0	0
16	SPA	0	0	0	0	1	0
17	SWE	1	0	1	1	0	1
18	UK	1	1	1	1	0	1

caseid　　国名の略号
wealthy　　一人あたりGNPが高いか、そうでないか
urban　　高度に都市化されているか、そうでないか
literate　　識字率が高いか、そうでないか
industrial　　工業労働者の割合が高いか、そうでないか
unstable　　政権が不安定か、そうでないか
survived　　戦間期に民主主義を維持できたか、そうでないか

[この節の例は、Rihoux and Ragin (2008)から取ったものである。]

C) 分析

fsQCAソフトの現在のバージョン（この原稿を執筆している時点2008年9月時点ではバージョン2.0）では、クリスプ集合の分析を行う方法として、次の2つのものがある。すなわち、Truth Table Algorithm（真理表アルゴリズム）とQuine(QCA 3.0)（クワイン）である。どちらの方法も、クワイン・マクラスキー法を使っており、同じ結果を導き出す。2つの方法の違いは、真理表をど

のように構築して管理するかというところにある。Truth Table Algorithmの方が簡明であるので、こちらの方法を使うことを推奨する。ここでは両方の方法を説明するが、推奨方法であるTruth Table Algorithmの方を最初に説明することにする。

真理表アルゴリズム

クリスプ集合のTruth Table Algorithm（真理表アルゴリズム）は、次の2つのタスクからなる。(1)原因条件の論理的に可能な様々な組み合わせについての、事例の分布の評価 (2)「当該組み合わせの原因条件を持つ事例が、結果が生じる事例の部分集合となる」という議論についての証拠の整合性の評価。

真理表アルゴリズムは、2つのステップの分析手続きからなっている。第1ステップは、素データから真理表のスプレッドシートを作成することである。ここでは主に、当該分析に含める原因条件と結果を特定する。第2ステップは、度数の閾値と整合度の閾値を選択し、真理表のスプレッドシートを分析しやすいように整えることである。これらのステップは、互いに関連を持って行われなければならず、それぞれの分析の際には両方のステップを実行しなければならない。

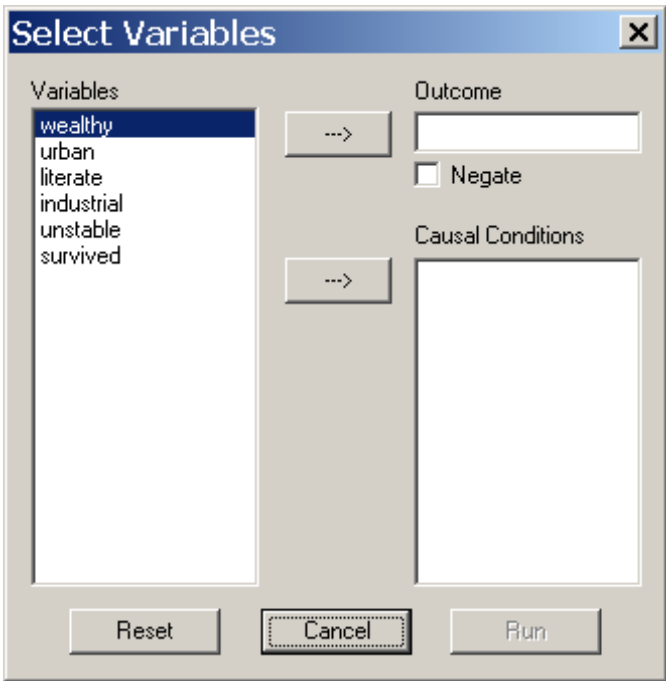
➢ 真理表のスプレッドシートを作成するには、次のように選択する。

Analyze（分析）

Crisp Sets（クリスプ集合）

Truth Table Algorithm（真理表アルゴリズム）

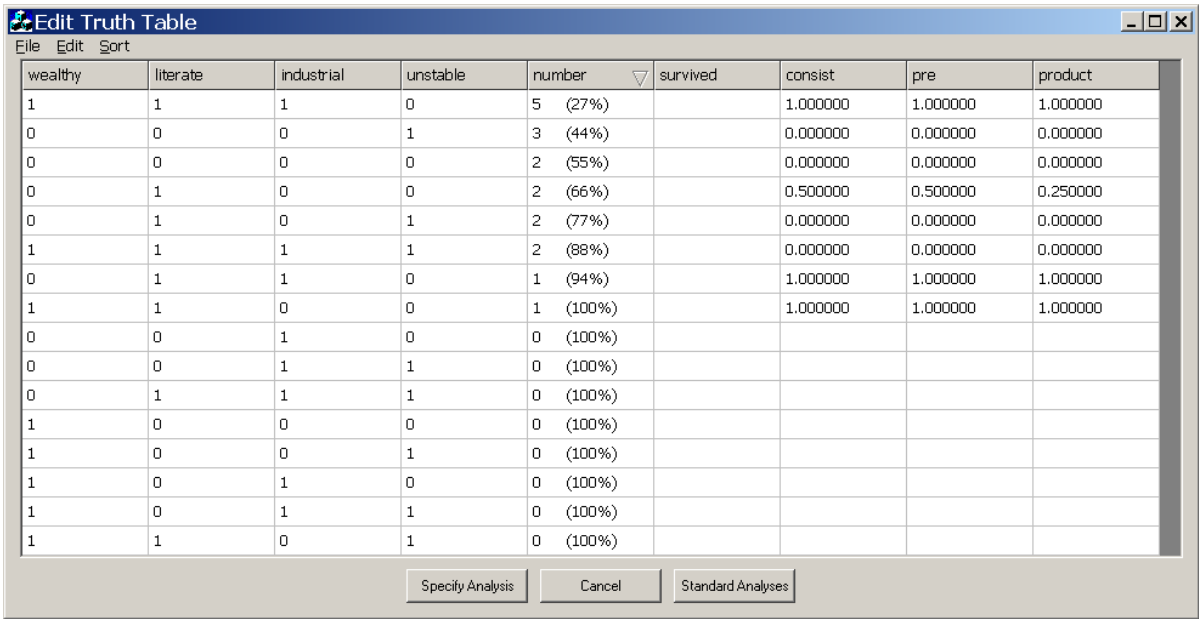
すると次のようなウィンドウが開き、ファイルの変数がリスト化されている。



- 説明したい変数を特定し、それをOutcome（結果）のフィールドに移す。

- 原因条件だと思われる変数を選択し、それをCausal Conditions（原因条件）のフィールドに移す。

- Run（実行）ボタンをクリックすると、次のようなウィンドウが現れ、完全な真理表が表示される。



wealthy	literate	industrial	unstable	number	survived	consist	pre	product
1	1	1	0	5 (27%)		1.000000	1.000000	1.000000
0	0	0	1	3 (44%)		0.000000	0.000000	0.000000
0	0	0	0	2 (55%)		0.000000	0.000000	0.000000
0	1	0	0	2 (66%)		0.500000	0.500000	0.250000
0	1	0	1	2 (77%)		0.000000	0.000000	0.000000
1	1	1	1	2 (88%)		0.000000	0.000000	0.000000
0	1	1	0	1 (94%)		1.000000	1.000000	1.000000
1	1	0	0	1 (100%)		1.000000	1.000000	1.000000
0	0	1	0	0 (100%)				
0	0	1	1	0 (100%)				
0	1	1	1	0 (100%)				
1	0	0	0	0 (100%)				
1	0	0	1	0 (100%)				
1	0	1	0	0 (100%)				
1	0	1	1	0 (100%)				
1	1	0	1	0 (100%)				

- 真理表は、2^k個（kは原因条件の数を表す）の行を持ち、これらの行は原因条件の全ての可能な組み合わせを表している。1という数字は、その条件を持つ事例の集合に完全に帰属することを表し、0という数字は、その条件を持つ事例の集合に全く帰属しないことを表す。それぞれの行について、以下のような変数の値が作成される。

number　当該原因条件の組み合わせを持つ事例の数

consist　真理表のそれぞれの行における、当該結果が生じる事例の割合[3]

pre　誤差減少率（PRE）に準ずる計算に基づいた、整合度の別の測定方法[4]（ファジィ集合の計算のために作られた）。クリスプ集合の分析では、consistと等しい。

product　consistとpreの積。クリスプ集合では、単にconsistを2乗したものになる。

結果の変数名の列（今の例ではsurvived）は空白になっていることに注意する。それぞれの条件

[3]（訳注） consist は consistency（整合度）の略。現在のバージョン（2009 年）では、raw consistency（粗整合度）と呼ばれている。

[4]（訳注） pre は proportional reduction in error（誤差減少率）の略。現在のバージョン（2009 年）では、PRI consistency（PRI 整合度）と呼ばれている。PRI は proportional reduction in inconsistency（不整合減少率）の略。具体的な計算方法は、95 ページで説明する。

の組み合わせについて、以下で説明する手続きを用いて結果を決定することは、研究者の仕事である。

- 研究者はまず、それぞれの条件の組み合わせ（行）を、その度数に基づいて、関連のある組み合わせと関連のない組み合わせに分類するルールを構築しなければならない。そのためには、numberの列に示されているそれぞれの行の事例数に基づいて、度数の閾値を選択すればよい。分析する事例の総数が比較的少ないときは、度数の閾値は1か2にすべきである。しかし総数が大きいときは、より大きな閾値が使われるべきである。原因条件の組み合わせにおける事例の分布を調べることが、非常に重要である。

- 条件の組み合わせ（行）は度数でソートできる。それには、numberの列の任意のセルをクリックして次のように選択する

  Sort（ソート）

  Descending（降順）

- 行をソートして度数の閾値を選択した後、閾値を満たさない行を全て削除する。行がnumberで降順にソートされている場合、閾値よりも下の最初の行をクリックして、次のように選択すればよい。

  Edit（編集）

  Delete current row to last row（現在の行から最後の行までを削除）

  行がソートされていない場合は、閾値を満たさない行をクリックして次のように選択することで、それらの行を個々に削除することができる。

  Edit（編集）

  Delete current row（現在の行を削除）

- 次のステップは、結果の部分集合となる条件の組み合わせと、そうでないものを区別することである。クリスプ集合でこれを決定するには、consistの列に表示される集合論的な整合性の測定値を使用すればよい。0.75以下の値は、あまり整合性がないことを表す。整合度の分布を評価するために、整合度を降順にソートするのが有用である（これは度数の閾値を満たさない行を削除した後に行うべきである）。ソートするには、consistの列の任意のセルをクリックして次のように選択すればよい。

  Sort（ソート）

  Descending（降順）

  上の方の整合度を見て、整合度の閾値を決定するのに使えるようなギャップを見つける。複数の閾値を検討したり、閾値の値を上げ下げして結果を評価したりすることが常に可能なことを念頭に置いておくこと。

- 次に、どの条件の組み合わせが結果の部分集合と見なせ、どれがそう見なせないのかを明らかにしなければならない。それぞれの条件の組み合わせの結果の列（今の例ではsurvived）のうちで、整合度の値が閾値以上となるもののところに1を入力する。それぞれの条件の組み合わせの結果の列のうちで、整合度の値が閾値を満たさないもののところに0を入力する。

- 別の方法として、Delete and code（削除とコーディング）の機能を使うと、今のプロセスを自動化できる。以下のように選択する。

  Edit（編集）

  Delete and code（削除とコーディング）

  1番目のフィールドでは、度数の閾値を選択する。デフォルトの数字は1となっているが、自分の選択した閾値をタイプすることで閾値を変えることができる。2番目のフィールドでは、整合度の閾値を選択する。デフォルトの数字は0.8となっているが、自分の選択した閾値をタイプすることで閾値を変えることができる。

  OKをクリックする。度数の閾値を満たさない行をプログラムが削除し、選択した整合度の閾値に応じて結果の列に0か1がコーディングされる。

- 以下のウィンドウは、次の2つの作業後の真理表を表している。
  1. 度数の閾値として1を適用し、観察されなかった条件の組み合わせ（8つ）を排除する。
  2. 整合度の閾値として0.9を適用し、0.9以上の整合度を持つ条件の組み合わせ（3つ）のsurvivedの列には1を入力し、0.9より小さい整合度を持つ条件の組み合わせ（5つ）のsurvivedの列には0を入力する。

Edit Truth Table

File Edit Sort

wealthy	literate	industrial	unstable	number	survived	consist	pre	product
1	1	1	0	5	1	1.000000	1.000000	1.000000
1	1	0	0	1	1	1.000000	1.000000	1.000000
0	1	1	0	1	1	1.000000	1.000000	1.000000
0	1	0	0	2	0	0.500000	0.500000	0.250000
0	1	0	1	2	0	0.000000	0.000000	0.000000
1	1	1	1	2	0	0.000000	0.000000	0.000000
0	0	0	0	2	0	0.000000	0.000000	0.000000
0	0	0	1	3	0	0.000000	0.000000	0.000000

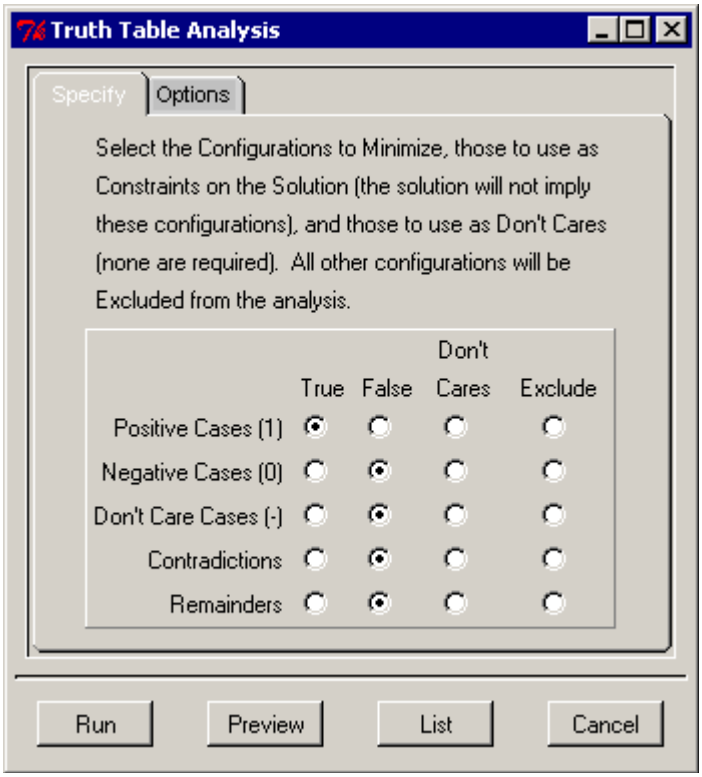
Specify Analysis　Cancel　Standard Analyses

ここから、単一の分析を指定するか、3つの「標準的な」分析（複雑、簡略、中間）を行うか、という2つの分析の可能性がある。Standard Analyses（標準分析）のボタン（3つの解を与える）をクリックするのが、推奨手続きである。

i) 分析指定オプション

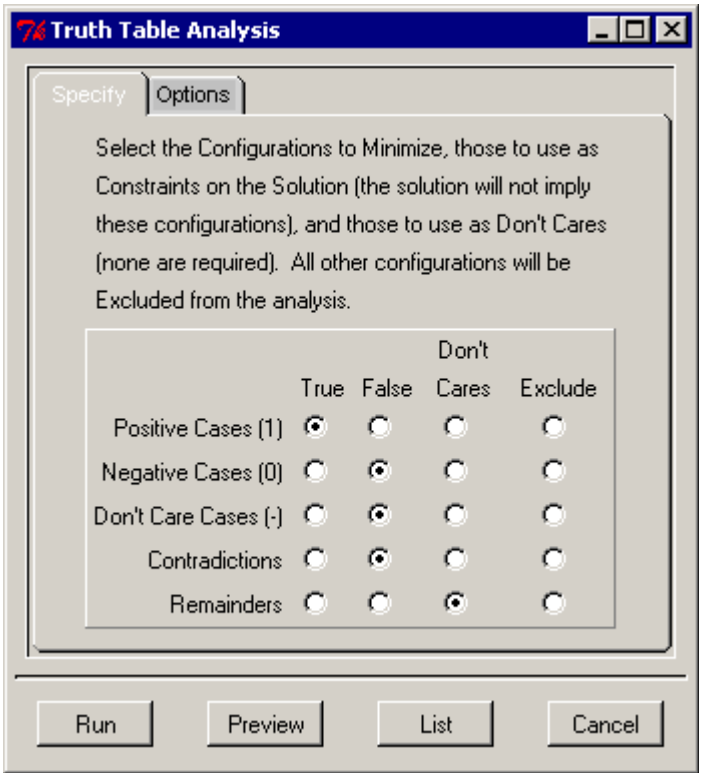
- 真理表を作成した後、Specify Analysis（分析を指定）を選択すると、Truth Table Analysis（真理表分析）ウィンドウが現れる。

- Specify（指定）パネル[5]において、Positive Cases（結果の値が1の事例）でTrueを選択し、他は全てFalseを選択すると、「最も複雑な」（most complex）解が導かれる。この場合、ウィンドウは次のようになる。



- 最も簡略な（most parsimonious）解を導くには、Positive Cases（結果の値が1の事例）でTrueを選択し、Negative Cases（結果の値が0の事例）、Don't Care Cases（結果の値が-の事例）、Contradictions（矛盾した事例）でFalseを選択し、Remainders（残余部分）でDon't Cares（ドント・ケア）を選択する。この場合、ウィンドウは次のようになる。

[5]（訳注） この Truth Table Analysis ウィンドウの Specify パネルでの設定の意味については、68 ページを参照。



- 他のオプション（主項や度数の出力に関するもの）は、Quineアルゴリズムのところで後述するのと同様の手続きに従って設定することができる。

- Preview（プレビュー）をクリックすると、出力ウィンドウに簡単な真理表が表示される。一般的に言って、Truth Table Algorithm（真理表アルゴリズム）を選択している場合、このボタンは役に立たない。なぜなら、ユーザーは前の作業で真理表を作成・コーディングしたばかりなので、それを「プレビュー」する必要はないからである。ともかく、Previewの手続きは行き止まりである（真理表分析は行われない）ので、Truth Table Algorithmを使うときはクリックするべきではない。

- List（リスト）をクリックすると、データ中にある原因条件の組み合わせとそのそれぞれの整合度のリストが得られる。これもTruth Table Algorithmの際には役に立つ手続きではない。なぜなら、ユーザーは前の作業で真理表を作成・検討したばかりだからである。Previewと同様に、この手続きは行き止まりである（真理表分析は行われない）ので、Truth Table Algorithmを使うときはクリックするべきではない。

- 分析を行うためには、Run（実行）のボタンをクリックする。すると、出力ウィンドウに結果が出力される。

ii) 標準分析オプション

- 真理表が完全にできあがったら、Standard Analyses（標準分析）を選択する。標準分析では、complex（複雑）、parsimonious（簡略）、intermediate（中間）の解が自動的に得られる。標準分析が推奨手続きである。なぜなら、intermediate solution（中間解）が得られるのは、この手続きだけだからである。中間解を導くために、ソフトウェアは、ユーザーからもたらされた情報をもとにして反事実分析（counterfactual analyses）を行う。

限られた多様性と反事実分析

比較研究の最も困難な面の1つは、研究者が扱うデータが比較的小規模なものであるという純然たる現実である。研究者はしばしば、「事例よりも変数の方が多い」という事態に直面する。比較研究を行う研究者が通常、事例の様々な側面の組み合わせ、すなわち事例の様々な側面がどのように組み合わさって構成されているか、に注目するという事実が事態を複雑にする。例えば、因果関係の議論について関心のある研究者が4つの原因条件を特定した場合、因果関係の議論を完璧に確かめるためには、これらの4つの条件の論理的に可能な組み合わせ16個全てを考慮するのが理想である。しかし、自然に発生する社会現象は、多様性が非常に限られてしまう。経験的世界が、社会科学者の議論に関連のある原因条件の論理的に可能な全ての組み合わせを提示してくれることは（下の表1の仮想的なデータに示されるように）ほとんどない。限られた多様性（limited diversity）は社会・政治現象の性質の中心をなすものだが、それらの分析を酷く複雑にするものでもある。

表1: 4つの原因条件（A, B, C, D）と1つの結果（Y）についての真理表

A	B	C	D	Y*
no	no	no	no	no
no	no	no	yes	?
no	no	yes	no	?
no	no	yes	yes	?
no	yes	no	no	no
no	yes	no	yes	no
no	yes	yes	no	?
no	yes	yes	yes	no
yes	no	no	no	?
yes	no	no	yes	?
yes	no	yes	no	?
yes	no	yes	yes	?
yes	yes	no	no	yes
yes	yes	no	yes	yes
yes	yes	yes	no	?

yes	yes	yes	yes	?

* Yの列で?がついているものは、事例がない。すなわち、結果が決定できない。

原因条件の組み合わせのうちで欠けているものの代替として、比較研究を行う研究者はしばしば、「思考実験」(Weber[1905]1949) を行う。すなわち、研究者は自らの理論的・実質的な知識を用いて、反事実的 (counterfactual) な事例を想像し、その結果についての仮説を立てる。QCAは事例横断的なパターンを評価するために真理表を用いるので、反事実的な事例（すなわち、現実には存在していない原因条件の組み合わせ) を考慮するプロセスは明示的でシステマティックである。実際、この特徴はQCAの主要な強みの1つである。しかし、反事実的な事例を明示的に考慮し、その結果を事例横断的なパターンについての主張に組み入れることは、社会科学にとって比較的新しいことである。したがって、QCAや反事実分析に関するベストプラクティスを特定することが必要不可欠である。

既存の理論に基づいて研究者が、原因条件A、B、C、Dが何らかの点で結果Yと結びついていると考えているとしよう。すなわち、これらの条件の（欠如ではなく）存在が、結果の存在と結びついていなければならない。経験的な証拠から、原因条件A、B、Cの存在とDの欠如がYと結びついている事例が多いことが示されたとする（すなわち、A*B*C*d→Y)。しかし、本当に重要なのは最初の3つの原因であるA、B、Cなのではないかと研究者は疑っている。A･B･CがYを生じさせるためには、Dが欠如している必要はない。しかし、A、B、Cと、Dの存在が組み合わさった事例は観察されなかった（すなわち、A*B*C*Dの事例は観察されなかった)。つまり、Dの欠如が結合因果 (A*B*C) の不可欠な一部であるかどうかを判断するための決定的な事例は、単に存在していないだけである。

反事実分析（すなわち、思考実験）を通して、研究者は、条件のこの仮説的な組み合わせ（A*B*C*D) は結果 (Y) を生じる可能性が高いと主張することができるかもしれない。つまり、研究者は、A*B*C*Dがもし存在すれば、Yを導くであろうと主張するかもしれない。この反事実分析によって、次のような論理式の簡単化ができる。

A*B*C*d + A*B*C*D → Y
A*B*C*(d + D) → Y
A*B*C → Y

この簡単化にはどれほどの妥当性があるだろうか？この質問に対する答えは、他の3つの原因条件（A*B*C）が存在する場合におけるDとYの結びつきについて、関連する理論的・実質的知識がどれぐらいあるかによる。もし研究者が既存の知識に基づいて、これらの条件の下において、Dの存在が結果Yが生じるための一因である（あるいは逆に、Dの欠如は一因ではない）と証明できるなら、前述の反事実的分析は妥当なものである。言い換えれば、既存の知識があれば、A*B*C*D → Yという主張は「容易な」反事実になる。なぜなら、そのような知識は、結果と結

びついていると考えられる条件の組み合わせ（A*B*C）に、余分な原因（D）を加えるものだからである。

QCAの強みの1つは、複雑解（complex solution）と簡略解（parsimonious solution）という両極端な解を導くための道具であるだけでなく、中間解（intermediate solutions）を特定するための道具ともなるということである。表1の真理表（p.57）をもう一度考えてみる。この表では、A、B、Cが原因条件で、Yが結果であった。前と同様、既存の理論的・実質的知識から、これらの原因条件の（欠如ではなく）存在が、結果と結びついていると仮定する。反事実を用いない分析の結果（すなわち、複雑解）からは、A*B*cの組み合わせがYを説明していることが明らかになった[6]。同じデータを用いつつも、より簡略な結果をもたらす反事実は何でも用いることを許した分析の結果（すなわち、簡略解）は、AだけでYの存在が説明できるというものであった。これら2つの結果を、両極端な解だと考えると、次のように直線上に図示できる。

A*B*c　　A

複雑解（A*B*c）は簡略解（A）の部分集合であることに注意しよう。これは、どちらの解もYが存在する場合の真理表の行をカバーするものでなければならないことを考えれば、論理的に当然のことである。なぜなら、簡略解は反事実の事例を残余部分として取り込んでいるので、追加的な行も含んでいるからである。複雑解と簡略解の直線の間には、同じ真理表に対する別の可能な解、例えばA*Bがある。これらの中間解は、簡略解を導いた残余部分の、様々な部分集合を結果に取り込んだ場合に生み出される。これらの中間解は、最も簡略な解（この例ではA）の部分集合であり、最も複雑な解（A*B*c）の超集合（superset）である。解の間の部分集合関係については、直線上に表すことができる。このことは次のことを意味している。すなわち、複雑解（A*B*c）で特定されている原因条件のうちの少なくともいくつかを用いるような組み合わせは、簡略解（A）で特定されている原因条件を含む限り、真理表の妥当な解となる。したがって今の場合、真理表の妥当な中間解は次のように2つある。

A*B

A*B*c　　A*c　　A

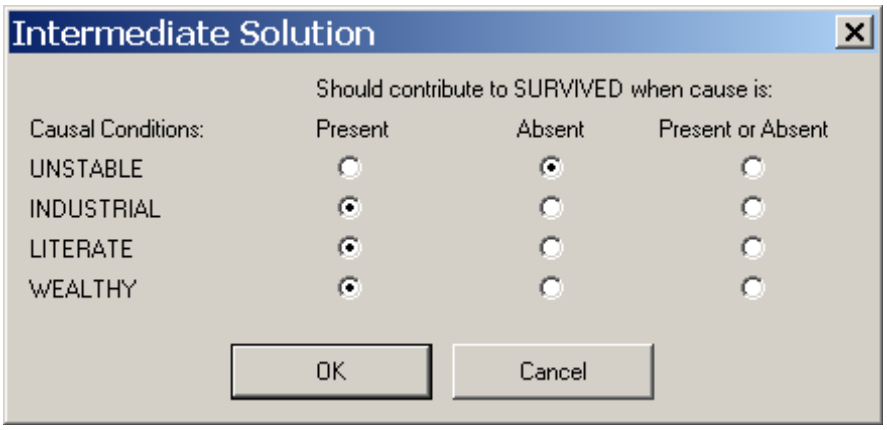
A*BとA*cという中間解はどちらも、簡略解の部分集合であり、複雑解の超集合である。1番目のA*Bは、A*B*CとA*B*cが結果Yと結びついているという反事実を許した場合である。2番目のA*cは、A*B*cとA*b*cという反事実を許した場合である。

これら2つの中間解のどちらが現実的かは、これらの解が取り込んだ反事実の妥当性による。1

[6]（訳注）ここでのcはCの否定、つまり原因条件Cの欠如を表している。このように、条件の文字の前に~を付けるのではなくて、条件の文字を小文字にすることにより、否定を表す方法もある。

番目の中間解が取り込んだ反事実は、「容易」なものである。なぜなら、これらの反事実によりA*B*cという組み合わせからcが取り除かれており、今の例では、Cの（欠如ではなく）存在が結果Yと結びついていることが既存の知識により支持されるからである。しかし、2番目の中間解が取り込んだ反事実は、「困難」なものである。なぜなら、これらの反事実によりA*B*cからBが取り除かれているからである。既存の知識によれば、Bの存在が結果Yの存在と結びついている。容易な反事実のみを取り込むべきであるという原則から、A*Bを最適な中間解として選択することが支持される。この解は、従来の事例指向的な研究者が、(1) 結果が存在するポジティブな事例に共通し（または少なくとも、ポジティブな事例の部分集合であり）、(2) 結果と結びついていると考えられ、(3) 結果が存在しないネガティブな事例には見られない、という特徴を持つ原因条件の組み合わせに対するストレートな関心に基づいて、データから導くものと同じである。

- Standard Analyses（標準分析）を選択した後、中間解の導出を補助するためのウィンドウが現れる。ここでは、研究者は、上で行ったのと同じように、それぞれの原因条件が理論的にどのように結果に影響を与えるかを選択しなければならない。当該原因条件が存在する場合に、その原因条件が結果が生じる一因となるのであれば、Present（存在）を選択する。当該原因条件が欠如している場合に、その原因条件が結果が生じる一因となるのであれば、Absence（欠如）を選択する。当該原因条件が存在しているかまたは欠如している場合に、その原因条件が結果が生じる一因となるのであれば、Present or Absent（存在または欠如）を選択する。



- 注意：主項を選択するアルゴリズムでは真理表を完全に縮約できない場合、Prime Implicantウィンドウ（主項ウィンドウ）が現れ、ユーザーが理論的・実質的知識に基づいて、使用する主項を選択しなければならない。このウィンドウが最も現れやすいのは、簡略解を導出する際であるが、3つの解のどれを導出する際にも起こりうることである（このウィンドウの説明は後述）。

- 分析を行うには、OKをクリックする。すると、中間解が出力ウィンドウに表示される。出力ウィンドウの中心の位置には簡略解が表示される。複雑解は簡略解のすぐ上に表示され、

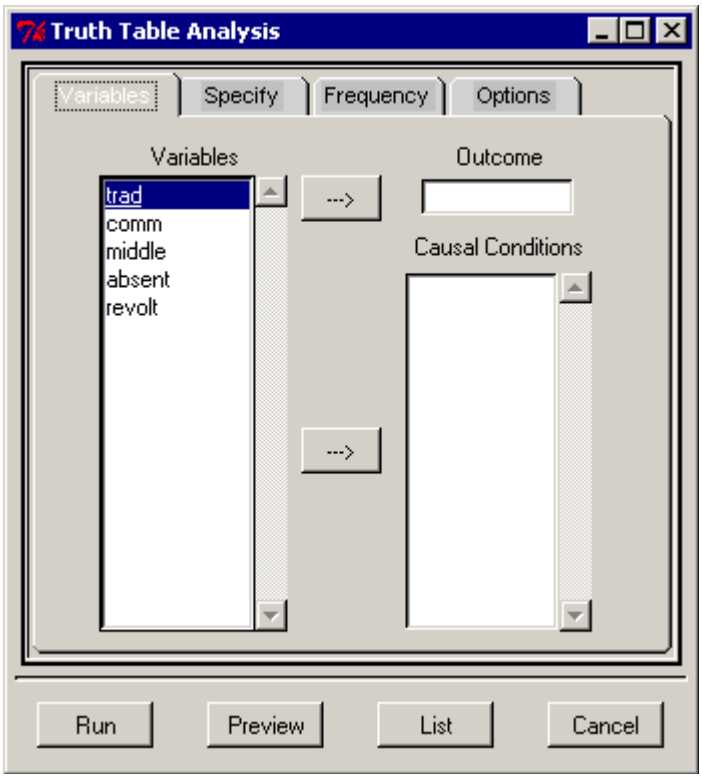
中間解は簡略解のすぐ下に表示されている。

Quineアルゴリズム（QCA3.0）

クリスプ集合のQuineアルゴリズムは、真理表アルゴリズムほど簡単ではない。なぜなら、Quineアルゴリズムでは、中間ステップとしてコーディング可能な真理表のスプレッドシートが提示されることがないからである。しかし、Quineアルゴリズムに慣れているユーザーのために、この機能はそのまま残されている（先に実装されたのはQuineアルゴリズムの方である）。

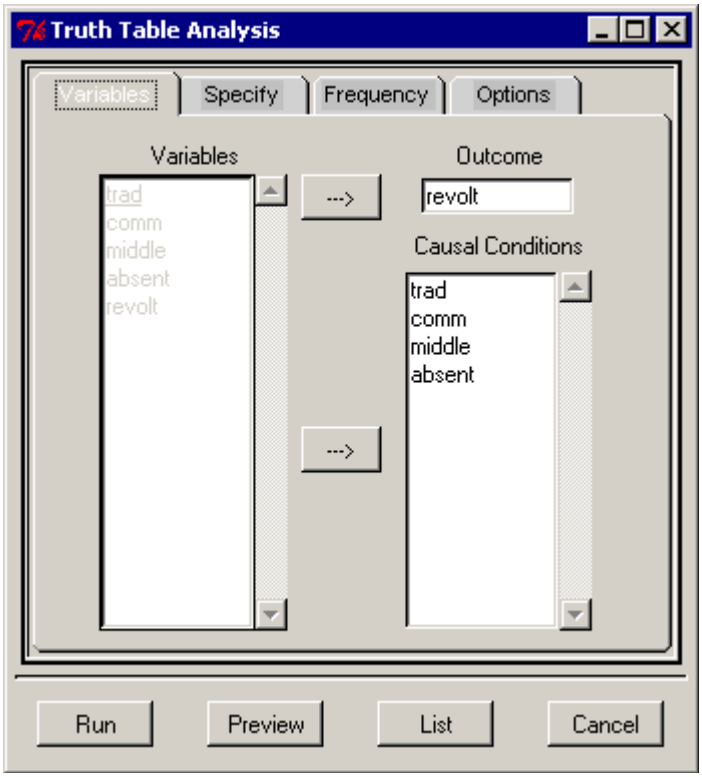
- Quineアルゴリズムでクリスプ集合のデータセットを分析するには、次のように選択する。
  Analyze（分析）
  Crisp Sets（クリスプ集合）
  Quine

- 次のようなウィンドウが開く（注：ハイライトされたボックスはVariablesボックスである。）



- 説明したい結果を特定し、それをOutcome（結果）のフィールドに移す。

- 残っている条件のうち結果を説明すると考えられるものをハイライトし、Causal Conditions（原因条件）のフィールドに移す。



真理表の行のリスト化

- 指定した真理表のリストを見るためには、次のように選択する。

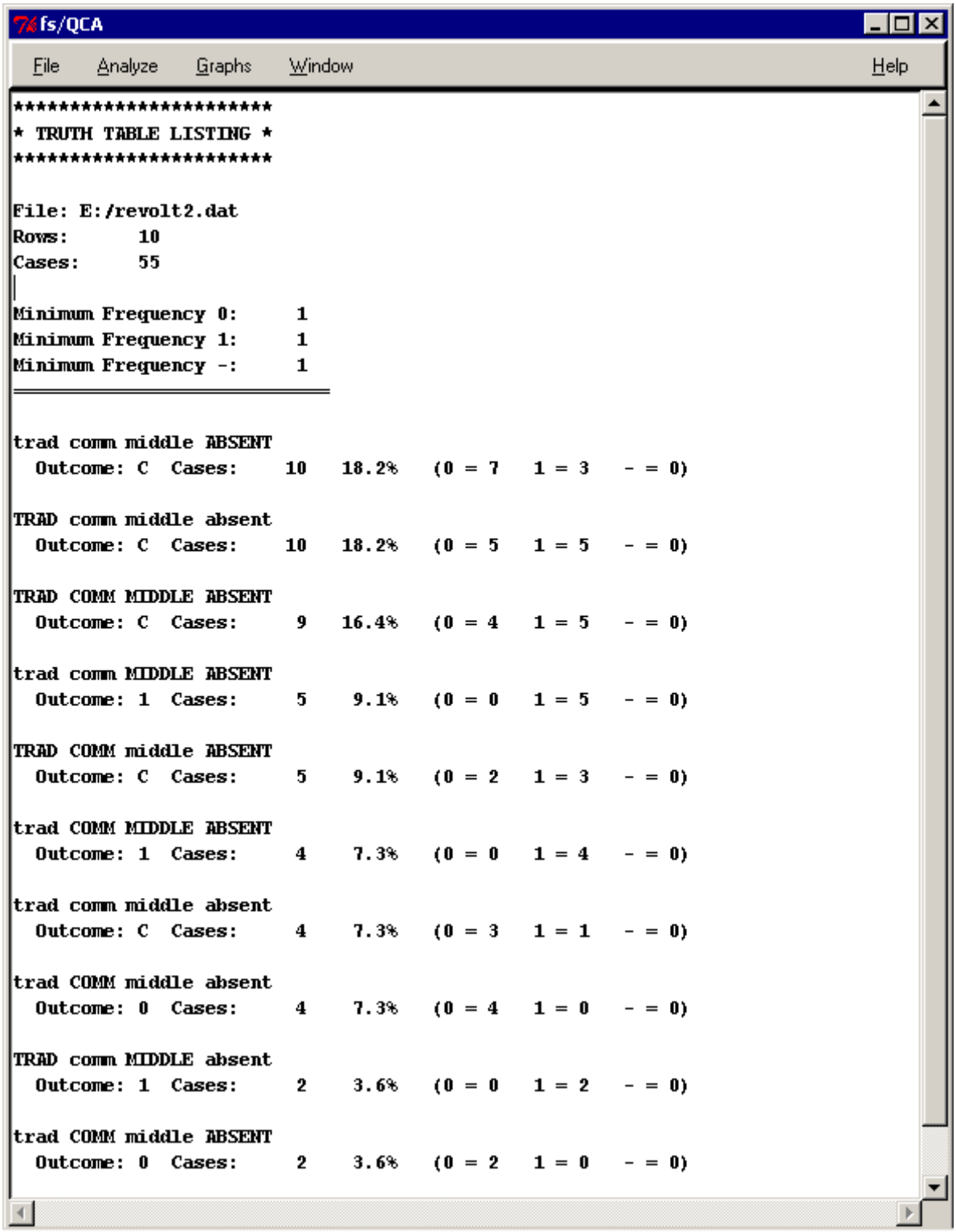
  Analyze（分析）

  Crisp-Sets（クリスプ集合）

  Quine

- 結果と原因条件を指定し、List（リスト）ボタンをクリックする。

- メインウィンドウに次のような出力が表示される。



[Ragin(1987:114)の表9と比較]

- １つ以上の変数が欠損値となっている事例は、真理表の条件の組み合わせには含まれていない。

- 条件の組み合わせは、事例の数によってソートされている。

- それぞれの行は、原因条件の1つの組み合わせを表示している。条件の欠如は小文字で表され、条件の存在は大文字で表される。

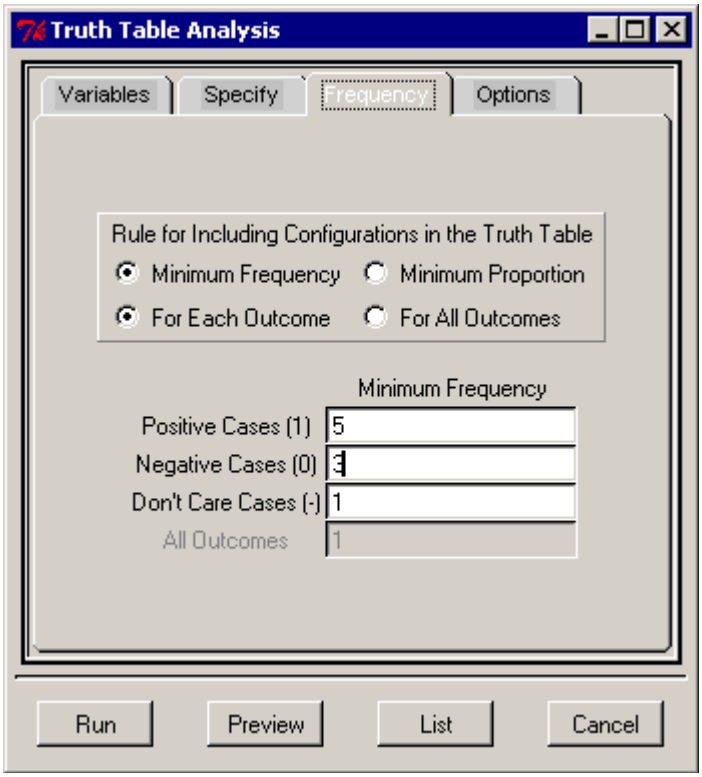
- その後には、当該条件の組み合わせの事例全ての、結果の状況が表示されている。上記の例では、5つの行ではっきりした結果となっている(4,6,8,9,10)。例えば、4番目の行は、5つの事例全てが結果1となっている。一方、8番目の行の4つの事例は、結果0である。ある条件の組み合わせの事例が全て同じ結果の値ではない場合、表では「C」という記号で矛盾を表す。上記の真理表では、5つの行で混合した、あるいは矛盾した結果となっている(1,2,3,5,7)。例えば、1番目の行の10個の事例は、そのうちの3つが結果1であり7つがそうではないので、矛盾した結果となっている。
- ここで（オプションとして)、真理表への算入に必要な、事例の最小度数または最小割合を指定することができる。

一般的に、このような証拠の評価は、クリスプ集合の真理表アルゴリズムで行った方が簡単である。

条件の組み合わせを算入するルール

- 真理表に条件の組み合わせを算入するルールを指定するには、次のように選択する。
  Analyze（分析）
  Crisp-Sets（クリスプ集合）
  Quine

- Frequency（度数）オプションをクリックすると、次のようなウィンドウが開く（注：ハイライトされたボックスはFrequencyボックスである)。

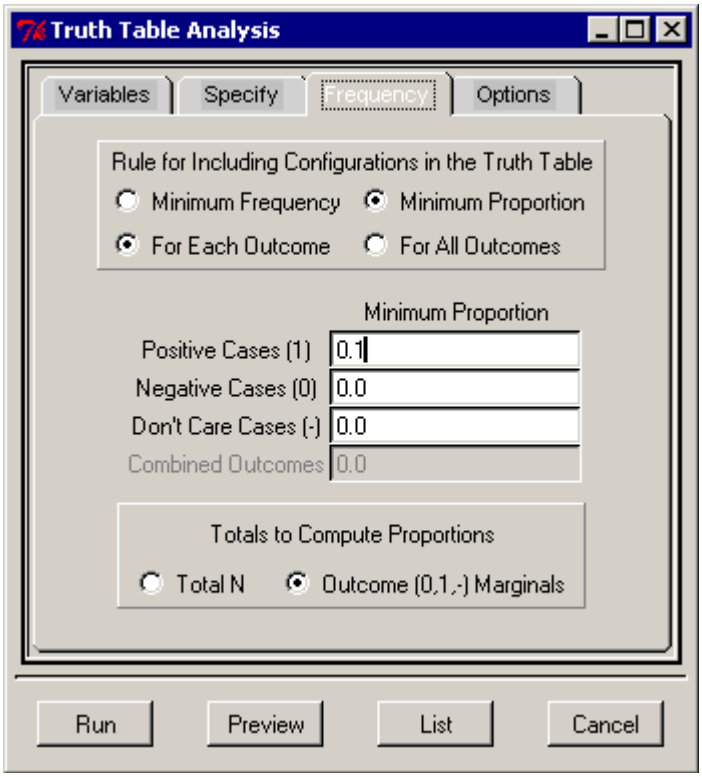


- それぞれの条件の組み合わせの事例の最小度数または最小割合を決定できる（上のウィンドウは最小度数のオプションを表示している）。

- 加えて、3つの結果の種類で異なった最小度数が必要かどうかを決定できる。上の例では、結果0の事例は少なくとも3つある場合、結果1の事例は少なくとも5つある場合のみに、条件の組み合わせを算入することを研究者は選択している。Don't Care（ドント・ケア）の結果となる条件の組み合わせは、Specify（指定）タブで取り除かれる。

- 3つの結果の種類全てで同じ最小度数を設定する場合、All Outcomes（全ての結果）を選択し、最小度数を設定する。

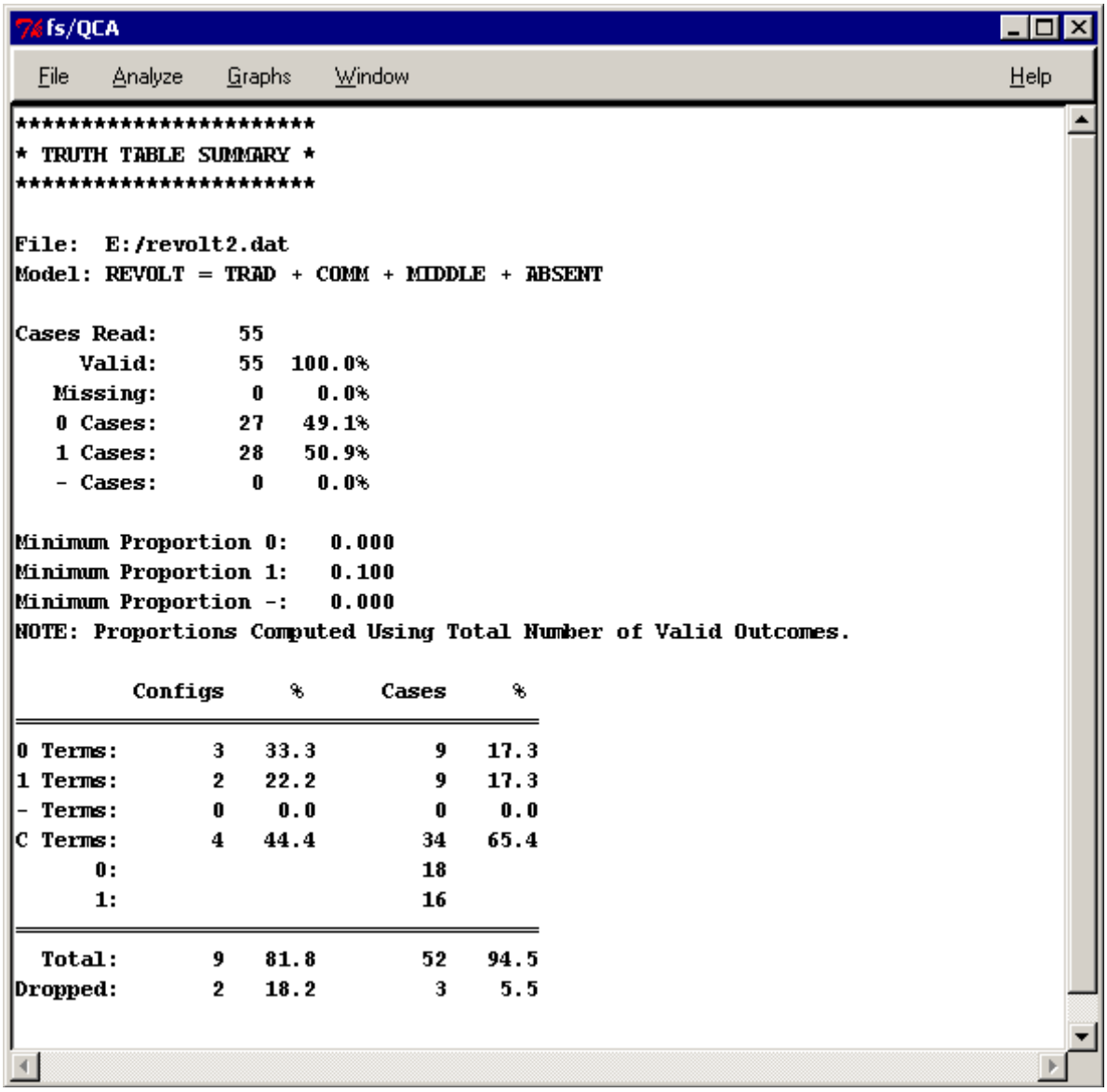
- それぞれの条件の組み合わせの事例の限界値を割合で設定する場合、Minimum Proportion（最小割合）オプションをクリックする。すると、次のようなウィンドウが現れる。



- 限界値をそれぞれの結果の割合とする場合、割合の分母を決定しなければならない。選択できるのは次のものである。
  - Total N（事例の総数）
  - Outcomes（結果）

  上の例では、事例の割合が少なくとも10%あるときにのみ、結果1の条件の組み合わせとして算入することを研究者は決定している。この章の例に適用すると、10%という限界値では、結果が1となる条件の組み合わせを全て排除してしまう。なぜなら、第4行と第6行の条件の組み合わせは、全事例のたった9.1%と7.3%であるからである（63ページの真理表を参照[Ragin(1987:114)の表9]）。

- 条件の組み合わせの算入のために設定した限界値を検討するには、Preview（プレビュー）ボタンをクリックする。すると、メインウィンドウに次のように表示される。



この出力の最初の2つの行は、データファイルの場所と、指定した原因条件・結果を表示している。

その後には、データセットの事例の要約が表示されている。例えば、データが完全な事例や欠損した事例の数・割合、そして結果の値が0、1、「ドント・ケア」である条件の組み合わせの数・割合といったものである。

その次の3つの行は、真理表への算入ための限界値の設定を示している（65ページの例の最小度数と比較）。

最後に、修正された真理表が示される。この表の最後の行では、研究者の指定によって排除された条件の組み合わせや事例の数・割合が示される（この例では、2つの条件の組み合わせで3つの

事例)。

簡単化のための条件の組み合わせの設定

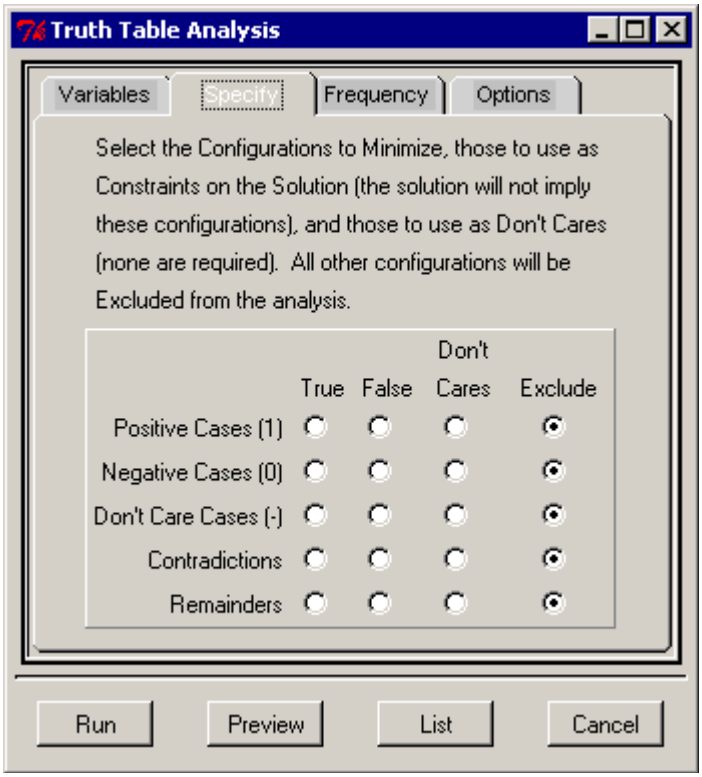
- 簡単化のための条件の組み合わせ（すなわち、解の制約として使用する条件の組み合わせや「ドント・ケア」として使用する条件の組み合わせ）を設定するには、次のように選択する。

  Analyze（分析）

  Crisp-Sets（クリスプ集合）

  Quine

- Specify（指定）オプションをクリックすると、次のようなウィンドウが現れる（注：ハイライトされたボックスはSpecifyボックスである)。



上記のウィンドウに示されたオプションを説明するには、それぞれの条件の組み合わせについての、ありうる様々な出力コードを明らかにしておく必要がある。あるデータセットの一部として次のような真理表があるとする。

	X_1	X_2	X_3	事例数	結果が生じた事例数	（出力コード）
行1	0	0	0	5	0	0
行2	0	0	1	8	8	1

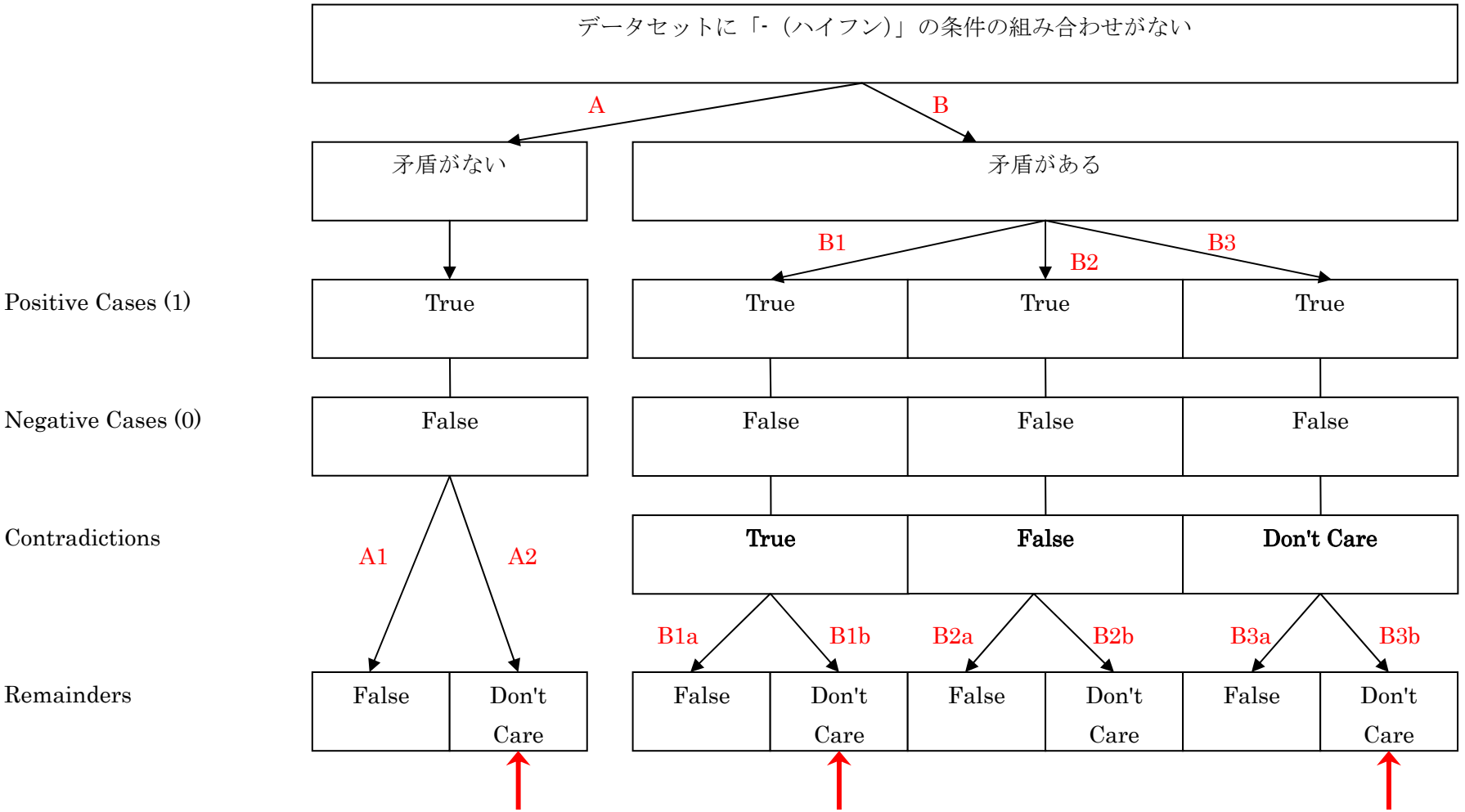
行3	1	1	0	1	-	-
行4	0	1	1	5	2	C
行5	1	1	1	0	?	?

それぞれの条件の組み合わせの出力コードは、次のうちのどれかとなる。

Positive Cases (1)（ポジティブな事例）	当該条件の組み合わせを持つ全ての事例で、結果が存在している場合（第2行参照）
Negative Cases (0)（ネガティブな事例）	当該条件の組み合わせを持つ全ての事例で、結果が欠如している場合（第1行参照）
Don't Care Cases (-)（ドント・ケアの事例）	当該条件の組み合わせを持つ全ての事例で、研究者が「ドント・ケア」の値をデータのスプレッドシートに割り振った場合。この場合、コンピュータは、状況に応じて、これらの条件の組み合わせを結果が存在しているものと扱うか欠如しているものと扱うことができる（第3行参照）。このコードはほとんど使われない。
Contradictions（矛盾）	全ての事例が同じ結果の値を持っているわけではない場合。例として、真理表の第4行は、5つの事例のうち結果が存在する事例は2つだけである（3つの事例は結果が欠如している）。
Remainders（残余部分）	データセット中に事例のない条件の組み合わせ（第5行参照）

Quineアルゴリズムによる簡単化では、「Exclude」と「False」は同じであることに注意すること。Quineアルゴリズムの作業で考慮されるのは、「True」と「Don't Care」の2つのみである。

クリスプ集合分析の指定のための８つの「最も一般的な」方法



オプション

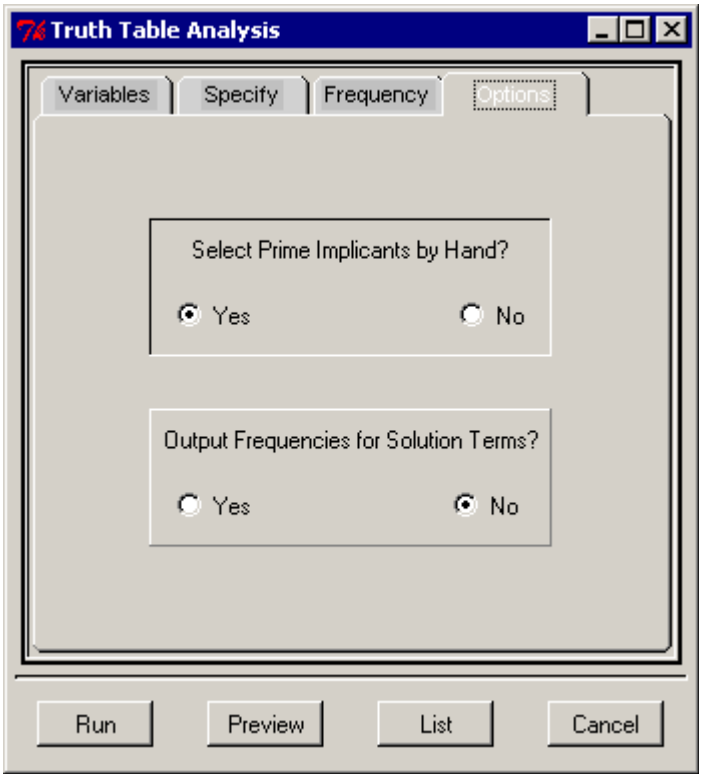
主項

主項とは、簡単化の規則（例: 1つの原因条件のみが異なる行は、同じ出力値を持つ場合、結合するという規則）を使って生み出した積項のことである。例えば、ABCとAbCは結合してACとなる。したがってACは、ABCとAbCという2つの原始的なブール項をカバーする主項である。言い換えれば、ABCとAbCはACの部分集合であり、ACはABCとAbCを包含する。

しかし、もとの原始的なブール項の全てをカバーするのに必要なよりも多くの主項がしばしば存在する。したがって、主項表を用いて、「論理的に結びついた」主項の中から選択を行うというオプションがある（主項の使用についてのより包括的な議論は、Ragin(1987:95)を参照）。

- このオプションを使用するかどうかを指定するには、次のように選択する。
  Analyze（分析）
  Crisp-Sets（クリスプ集合）
  Quine

- Options（オプション）をクリックすると次のようなウィンドウが開く（注: ハイライトされたボックスはOptionsボックスである）。



- 論理的に同等な（すなわち、真理表の同じ行をカバーする）主項の中からの選択を可能にす

るには、Select Prime Implicants by Hand?（主項を手動で選択しますか？）という項目でYESを選択する。NOを選択すると、可能な選択の全てが解の中で表示される。すなわち、部分的に縮約された主項の表が、「次のうちの1つ」という見出しの下に選択肢として出力される。

主項表を使用して解の項を選択

- ユーザーが主項を手動で選択することにした場合、選択できる論理的に同等な主項がある場合に限り、Prime Impricant Chart（主項表）ウィンドウが現れる。

  - 主項を選択するにあたって、解に必ず現れる必要不可欠な主項（真理表のある特定の行をカバーをしている主項がそれしかないようなもの）から始まって、さらなる単純化が不可能なところまで、表の縮約が試みられるというアルゴリズムをプログラムは採用している。このアルゴリズムで表を完全に縮約できなかった場合に、理論的・実質的知識に基づいて、使用する主項をユーザーが選択することができる。

  - 主項表には2つのタブがある。Solution（解）タブを選ぶと、解に必ず含まれる必要不可欠な主項が表示される。このフィールドが空白の場合、必要不可欠な主項は存在しない。

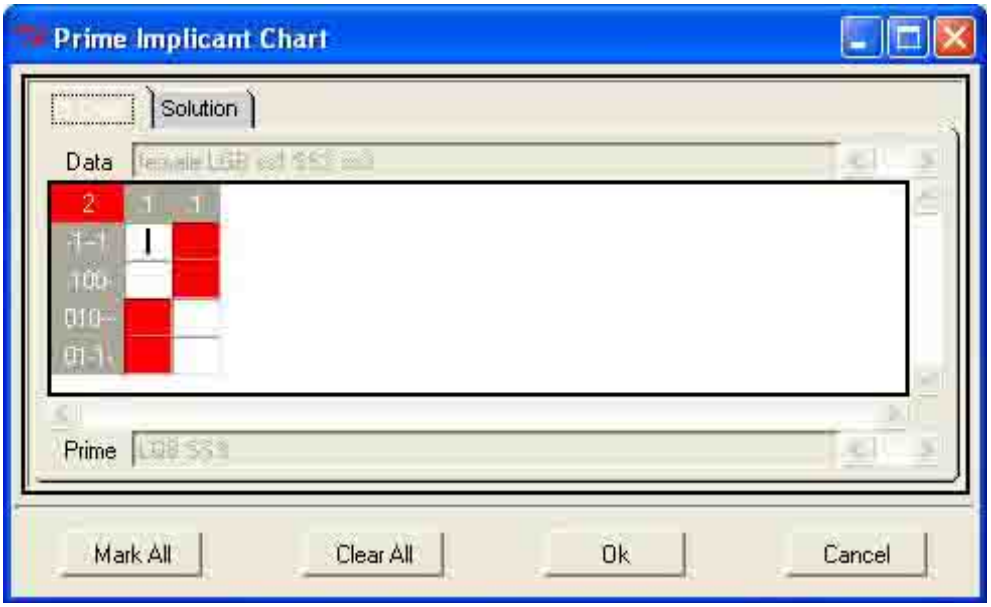
  - PI Chart（主項表）タブでは、ユーザーが選択できる主項が表示される。表のそれぞれの列は、1つ以上の主項でカバーされる真理表の行を表している。一番上のData（データ）フィールドは、問題となっている真理表の行（カバーされる必要のある行）を表示する。

  - Prime（主項）フィールドは、ユーザーが選択できる主項を表示する。表のそれぞれの行は、選択できる主項のうちの1つを表している。表の内部をクリックすると、そのセルの列と行の情報を表示する。

  - 主項を選択するには、最初の列のセルをクリックすればよい。このセルは、0, 1, -の組み合わせでその主項を表現している。選択を変更する場合には、Clear All（全てクリア）をクリックすれば、主項を選択していない状態にできる。Mark All（全て選択）をクリックすると、全ての主項を選択できる。

  - 表の左上の角の数字は、何個の主項を選択しなければならないかを示している。列の一番上の数字は、真理表の当該行につき何個の主項を選択しなければならないかを示している（この主項表の列は、カバーしなければならない真理表の行を表していることを想起すること）。適切な主項が選択されると、左上の角の数字も列の一番上の数字も

1だけ減る。複数の主項（主項表の列）で表される真理表の行がある場合、それぞれの行について少なくとも1つの主項を選択しなければならない。それぞれの主項（主項表のそれぞれの行）がカバーしているセルは、赤で色付けされている。その主項が選ばれると、カバーしているセル（真理表の行）の色が灰色に変わる。全ての該当する真理表の行について1つ以上の主項が選ばれると、左上の角の数字が0になり色が緑になる。



上の主項表ではユーザーは、1番目と3番目の主項、1番目と4番目の主項、2番目と3番目の主項、2番目と4番目の主項を選ぶことができる。もちろん、追加で主項を選ぶこともできるが、これらの4つの組み合わせが、真理表の残った行（主項表の列で表される）をカバーする最小の選択である。

解の項の度数

- 解の項の度数を出力するかどうかを指定するには、次のように選択する。
  Analyze（分析）
  Crisp-Sets（クリスプ集合）
  Quine

- Options（オプション）をクリックして、Select Output Frequencies Solution Terms（解の項の度数の出力の選択）の欄でYESかNOかを選択する。

このオプションを選択すると、クリスプ集合の解のそれぞれの項についての、事例の数が表示される。例えば、下記の解の項では、traditionalの欠如とmiddleの欠如という原因条件の組み合わ

せがあって結果が生じたものは、データセットの事例のうち12個である。

traditional middle (12) +
traditional ABSENTEE (5) +
COMMERC ABSENTEE (24) +
TRADITIONAL commerc absentee (14)

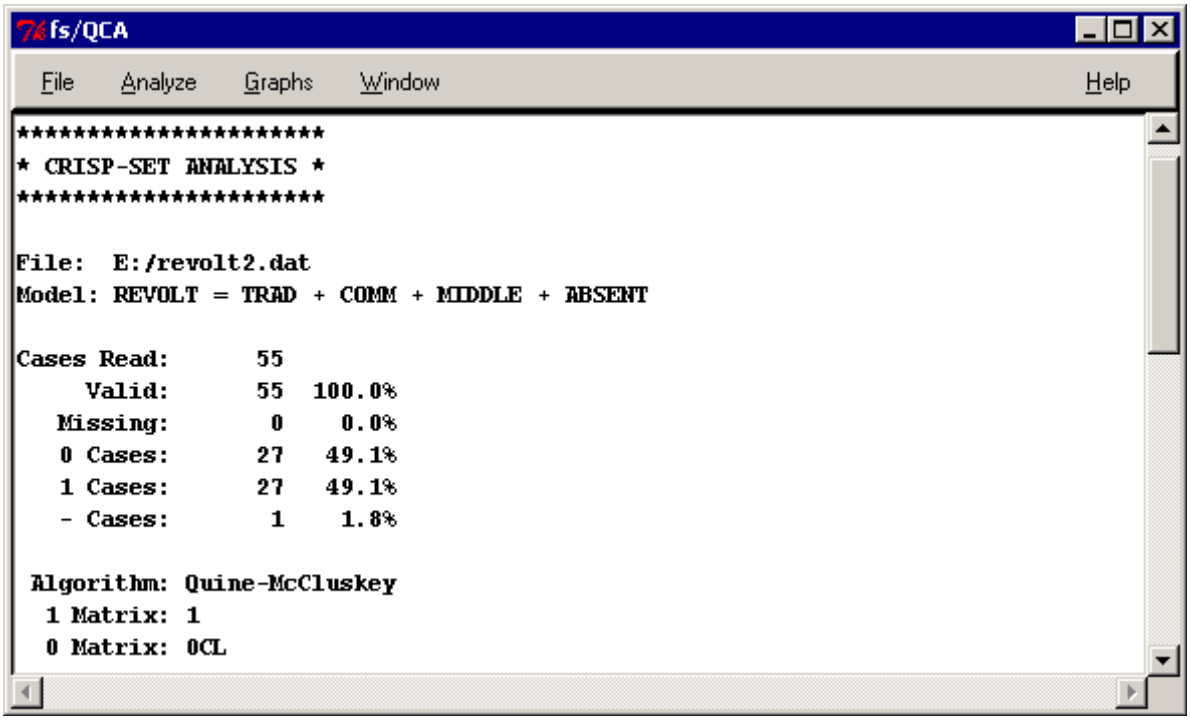
この情報は、真理表アルゴリズムでも、解の項のcoverage（被覆度）の計算によって分かる。

D）出力（クリスプ集合）

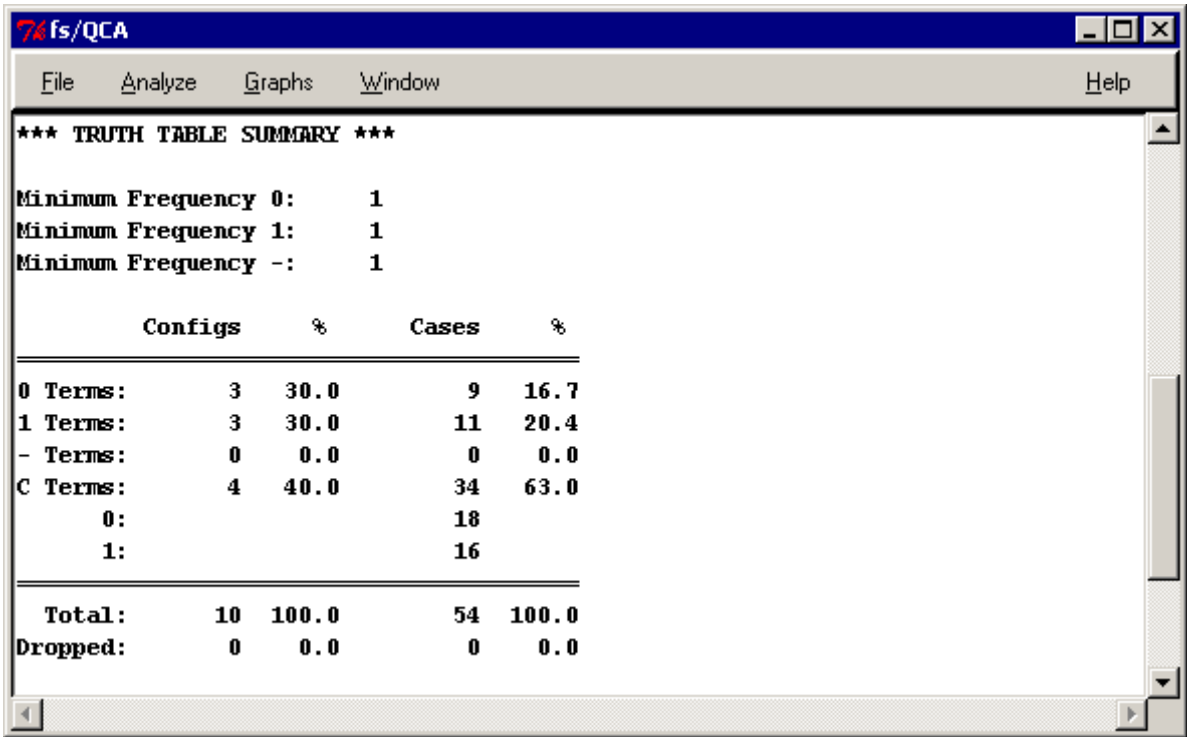
各種のオプションを指定して真理表を簡単化したら、次のような出力がメインウィンドウに表示される。

Quineアルゴリズムでの出力

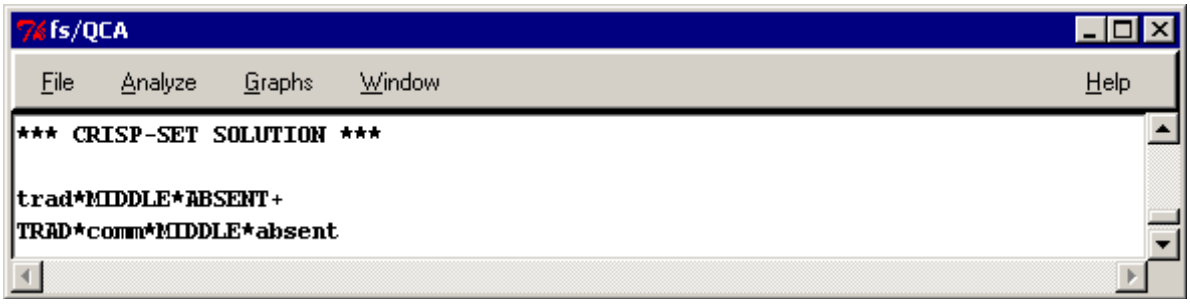
出力の最初の部分は、使用したデータを表している。最初の2つの行は、ファイルのディレクトリと指定したモデルを表している。その次は、真理表の要約であり、各種類の条件の組み合わせの事例の数とパーセントが表示される。その後に、使用したアルゴリズムの種類、簡単化した条件の組み合わせ、設定した制約が表示される。下の例では、結果が存在する条件の組み合わせが簡単化されており、結果が欠如している条件の組み合わせ、矛盾、残余部分が制約として設定されている。



真理表の要約では、条件の組み合わせのどれかを除いたかどうか、それぞれの条件の組み合わせに何個の事例があるか、何個の事例を分析のために残したかがわかる。次の例では、除いた条件の組み合わせはない。



出力の最後の部分では、クリスプ集合分析の解が導出されている。



この例では、(1) 農民の伝統主義の欠如と農民の中間層の存在と不在地主の存在 (2) 農民の伝統主義の存在と商業主義的農業の欠如と農民の中間層の存在と不在地主の欠如 のいずれかの組み合わせの場合に、反乱が生じる。

真理表アルゴリズムでの出力

真理表を指定して簡単化すると、メインウィンドウに出力が表示される。出力の最初の部分にはデータが表される。最初の2つの行は、ファイルのディレクトリと指定したモデルを表している。次の行では、分析のために読み込んだ真理表の行の数をリスト化している。その後に、使用

したアルゴリズムの種類、簡単化した条件の組み合わせが表示される。

最初に提示される解は常に複雑解であり、その後に簡略解、中間解が続く。3つの解は全て見出しが付けられている。度数と整合度の区切り値（cutoff）も表示される。指定した区切り値（例えば、真理表のスプレッドシートのdelete and code機能で使用したもの）を表示するのではなく、実際に解を導くのに使用した条件の組み合わせの中で最小の度数や最小の整合度が表示される。例えば、整合度の区切り値として0.8を選択したが、実際の整合度には0.9と0.7の間にギャップが存在する場合、0.9が整合度の区切り値として表示される。なぜなら、true（1.0）とコーディングされた行の中で最小の整合度は0.9だからである。複雑解は次のようになる。

```
File: C:/Documents and Settings/Desktop/benoit.csv
Model: SURVIVED = f(WEALTHY, LITERATE, INDUSTRIAL, UNSTABLE)
 Rows: 8
 Algorithm: Quine-McCluskey
      True: 1
--- COMPLEX SOLUTION ---
frequency cutoff: 1.000000
consistency cutoff: 1.000000
                                    raw      unique
                                  coverage   coverage    consistency
                                ----------  ---------- -----------
WEALTHY*LITERATE*unstable+       0.750000    0.125000    1.000000
LITERATE*INDUSTRIAL*unstable     0.750000    0.125000    1.000000
solution coverage: 0.875000
solution consistency: 1.000000
```

解の見出しの後に、度数と整合度の区切り値が表示される。Specify Analysis（分析を指定）を選択した場合、Truth Table Solution（真理表の解）という項目が1つだけ表示される。Standard Analyses（標準分析）を選択した場合、複雑（complex）、簡略（parsimonious）、中間（intermediate）の解が導出され、見出し付きで表示される。さらに中間解では、ユーザーがダイアログボックスで指定した原因条件の仮定のリストも表示される。

解では、1行ずつそれぞれ別々の因果経路が表示される。今の例では、原因条件の2つの組み合わせが結果の存在と結びついている。すなわち、富裕で識字率が高く政権が安定しているか、識字率が高く工業労働力人口の割合が高く政権が安定しているならば、民主主義が継続する。同時に表示されている整合度（consistency）と被覆度（粗被覆度raw coverageと固有被覆度unique coverageを含む）についての概略は後述するが、より詳しくはRagin(2008)で説明されている。

今の例の真理表の簡略解は次のようになる。

```
File: C:/Documents and Settings/Desktop/benoit.csv
Model: SURVIVED = f(WEALTHY, LITERATE, INDUSTRIAL, UNSTABLE)
Rows: 8
Algorithm: Quine-McCluskey
True: 1-L
--- PARSIMONIOUS SOLUTION ---
frequency cutoff: 1.000000
consistency cutoff: 1.000000
                            raw        unique
                          coverage     coverage     consistency
                         ----------   ----------  -----------
WEALTHY*unstable+        0.750000     0.125000     1.000000
INDUSTRIAL*unstable      0.750000     0.125000     1.000000
solution coverage: 0.875000
solution consistency: 1.000000
```

前と同様に、使用したアルゴリズムの種類、簡単化した条件の組み合わせが表示される。解でも前と同様、2つの因果経路が示される。すなわち、富裕であり政権が安定しているか、工業労働力人口の割合が高く政権が安定している場合、民主主義が継続する。

通常、中間解が最も解釈しやすいが、簡略解はポジティブな事例とネガティブな事例を区別するために必要不可欠な条件はどれかを示す。

5. ファジィ集合分析

マニュアルのこのパートでは、**ファジィ集合**の分析を扱う。これは、Ragin(2000)やRagin(2008)で詳しく議論されている。帰属と非帰属という2つの互いに排他的な状態のみ認める代わりに、0から1の範囲のメンバーシップ度（帰属度）を認めることでクリスプ集合を拡張したのが、ファジィ集合である。ファジィ集合を構築する方法は多数ある。次の3つの方法が一般的である。

4値ファジィ集合（0, 0.33, 0.67, 1）
6値ファジィ集合（0, 0.2, 0.4, 0.6, 0.8, 1）
連続ファジィ集合（0以上1以下の任意の値）

inclusion（包摂）アルゴリズムとtruth table（真理表）アルゴリズムという、2つのファジィ集合のアルゴリズムがある。包摂アルゴリズムはRagin(2000)、真理表アルゴリズムはRagin(2008)やRihoux and Ragin(2008)で説明されている。包摂アルゴリズムは、より頑健で真理表アルゴリズムとも整合的なものにするために、現在見直し中である。したがって、包摂アルゴリズムへのアクセスは現在ブロックされている。真理表アルゴリズムはより頑健であることが分かっており、現在のところの推奨アプローチである。

A）ファジィ集合の演算

論理積（「かつ」）や論理和（「または」）という論理演算はファジィ集合のアルゴリズムでも使用されるが、クリスプ集合での使用とは異なっている。以下は、論理積や論理和など一般的な演算の入門である。

論理積

ファジィ集合では論理積は、交わった集合内のそれぞれの事例のメンバーシップ度の最小値を取ったものである。例えば、ある国の、貧しい国の集合でのメンバーシップ度が0.34で民主主義国でのメンバーシップ度が0.91のとき、貧しくかつ民主的な国の集合でのその国のメンバーシップ度は、2つのメンバーシップ度のうち小さい方である0.34となる。

論理和

2つ以上の集合は、論理和によって接合し和集合になることができる。例えば、「先進国」と「民主主義国」という2つの異なる条件が、何らかの結果（例: 官僚主義的政府）の共通した基盤になっているという推測の下、研究者が「先進国または民主主義国」という集合に興味を持つかもしれない。伝統的には、「先進国または民主主義国」（一方または両方の特徴を持つ国）の完全なリストを作るためには、クリスプ集合のカテゴリーが使われるだろう。それに対しファジィ集合では、構成要素たる集合でのそれぞれの事例のメンバーシップ度の最大値に注目する。すなわち、2つ以上の集合の和集合でのメンバーシップ度は、その和集合を構成するそれぞれの集合での事例のメンバーシップ度の最大値である。したがって、ある国の、民主主義国の集合でのメンバーシップ度が0.15で先進国でのメンバーシップ度が0.93のとき、「民主主義国または先進国」の集合でのその国のメンバーシップ度は0.93である。

否定

クリスプ集合と同様、ファジィ集合でも否定を取ることができる。クリスプ集合では、否定はメンバーシップ度を1から0、0から1へと切り替えるものだった。ファジィ集合でも、この単純な数学的法則は維持される。ただ、値は0と1というブール値から、0から1までの値に拡張される。ファジィ集合Aの否定についての、事例のメンバーシップ度を計算するには、次のように単に1からそのメンバーシップ度を引けばよい。

Aの否定でのメンバーシップ度 = 1- (Aでのメンバーシップ度)

これは、$\sim A_i = 1 - A_i$ という式で表される。この式の添え字のiはi番目の事例、~A はAの否定、~という記号は否定を意味する。例えば、アメリカの「民主主義国」の集合でのメンバーシップ度が0.79であるならば、「非民主主義国」の集合でのメンバーシップ度は0.21となる。

B) ファジィ集合、必要条件、十分条件（ファジィ部分集合関係）

クリスプ集合での部分集合の法則やメンバーシップ度の算術的関係

国家の崩壊は社会革命の必要条件ではあるが十分条件ではないという例（Ragin 2000:211）を考えてみよう。原因が結果の必要条件ではあるが十分条件ではない場合、結果の事例の集合は原因の事例の集合の部分集合を構成するということが論理的に言える。部分集合関係を理解する別の方法は、クリスプ集合のメンバーシップ度（1と0）の間の算術的関係の観点から見ることである。結果の事例の集合が原因の事例の集合の部分集合である場合、結果のブール値（1と0）は原因のブール値以下である。

ファジィ集合での部分集合の法則やメンバーシップ度の算術的関係

ファジィ集合では、結果が生じた国を「選択する」（クリスプ集合での必要条件分析における通常の最初のステップ）ことは困難である。なぜなら、社会革命の集合でのメンバーシップ度の大きさは、国ごとに異なっているからである。同様に、原因条件（国家の崩壊）に関しての事例の一致を評価することも非常に困難である。なぜなら、この集合でもメンバーシップ度の大きさは事例ごとに異なっているからである。

幸い、部分集合の法則やメンバーシップ度の算術的関係は、ファジィ集合でも維持される。ファジィ集合では、集合Aでのそれぞれの事例のメンバーシップ度が集合Bでの同じ事例のメンバーシップ度以下である場合、集合Aが集合Bの部分集合である。さらに、結果におけるメンバーシップ度が原因条件におけるメンバーシップ度以下である場合、結果が発生した事例の集合は、当該原因条件を持つ事例の集合の部分集合である。図1は、2次元の座標でこの算術的関係を表している。研究者はこのパターンを見つけた場合、必要条件の因果関係の議論を支持する証拠として援用するだろう。

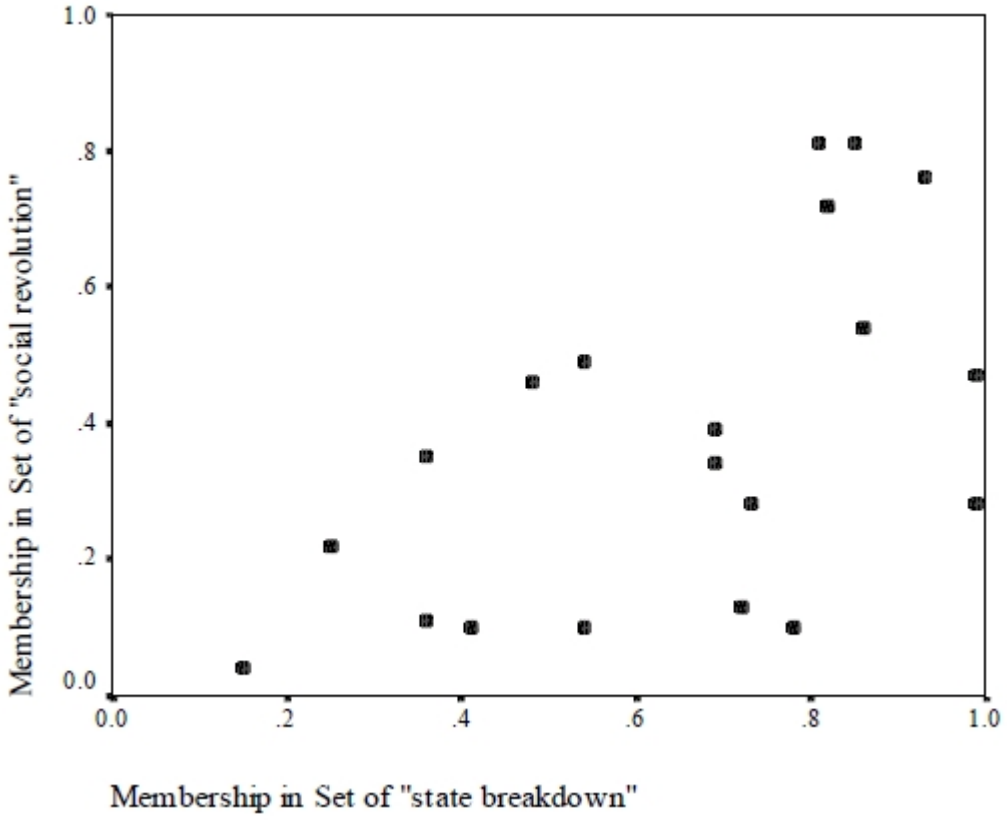


図1:　「国家の崩壊」と「社会革命」のプロット

十分条件の評価は、当該原因条件を持つ事例の集合が、結果が発生した事例の集合の部分集合であるかどうかを確かめることだと見ることができる。上の場合と同様、部分集合関係を理解する別の方法は、メンバーシップ度の間の算術的関係の観点から見ることである。原因条件やその組み合わせが結果の十分条件であると議論するためには、原因条件におけるファジィ集合のメンバーシップ度が、結果におけるメンバーシップ度以下でなければならない。

Ragin(2000:236ff)から取った、次のような例を考えてみよう。図2は、十分条件となっている原因条件の組み合わせ（~crossclass*~multiracial）とその結果（ideological conflict）の間の算術的関係を表している。図2の上側の三角形状のプロットは、「人種が均一かつ階級が均一」のファジィ集合におけるメンバーシップ度が、「イデオロギー対立」のファジィ集合におけるメンバーシップ度以下であるという事実を、直接的に表している。

十分条件関係を評価する際と必要条件関係を評価する際の部分集合の法則の適用には、重要な違いがあることに注意が必要である。必要条件関係を証明するには、結果が原因の部分集合であることを研究者は示さなくてはならない。十分条件関係を証明するには、原因が結果の部分集合であることを研究者は示さなくてはならない。

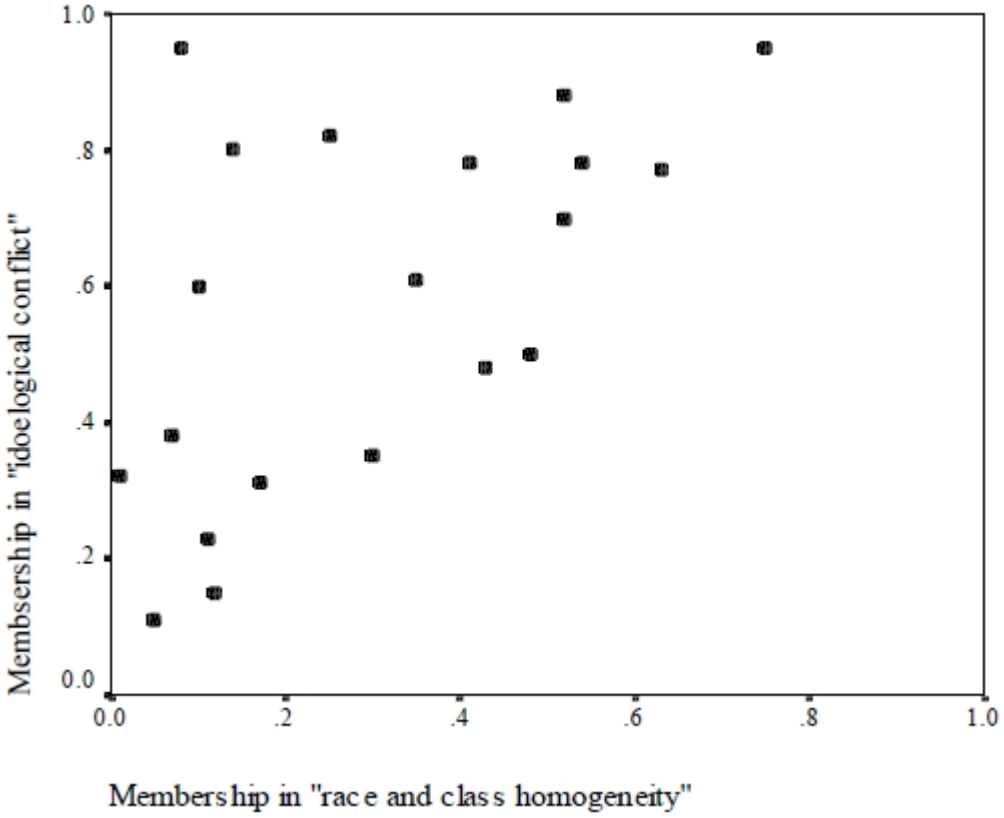


図2:　「人種が均一かつ階級が均一」と「イデオロギー対立」のプロット

C) ファジィ集合の真理表アルゴリズムの使用

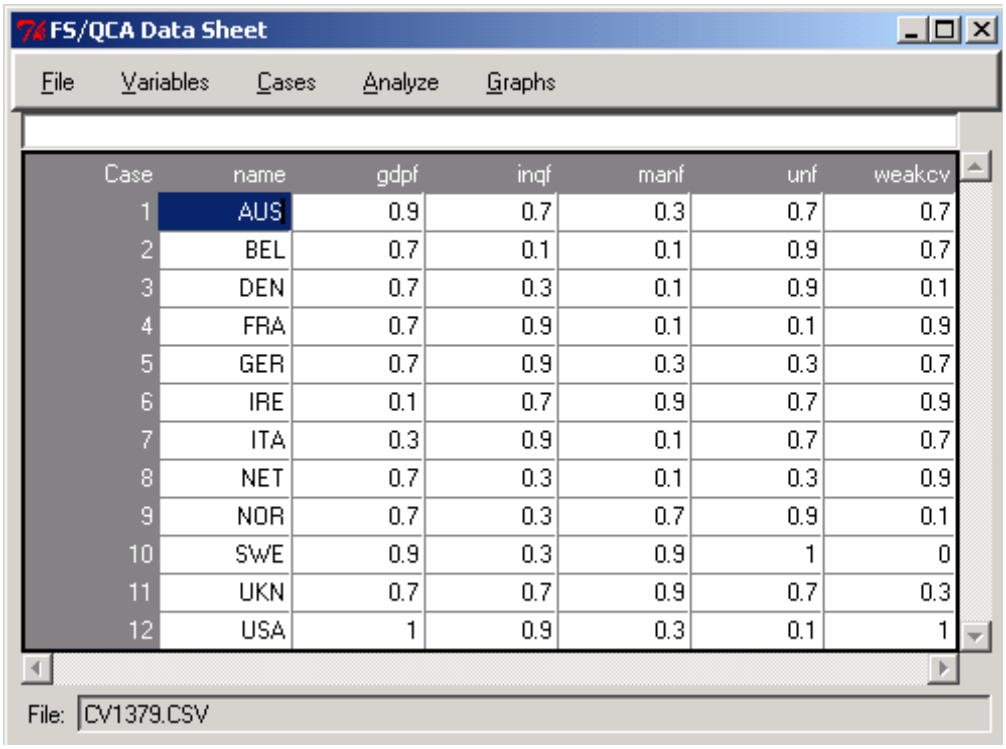
真理表を用いたファジィ集合の分析の方法は、fs/QCAのバージョン2.0で導入された。この方法については、Ragin(2008)やRihoux and Ragin(2008)で詳しく説明されている。

ファジィ集合の真理表アルゴリズムは、3つの柱を架橋するものとして概念化することができる。第1の柱は、クリスプ集合の真理表の行と、ファジィ集合の原因条件で定義されたベクトル空間の端[7]（corner）の間に存在する直接的な対応関係である(Ragin 2000)。第2の柱は、論理的に可能な様々な原因条件の組み合わせ（すなわちベクトル空間の端）についての、事例の分布の評価である。ベクトル空間のある端は強いメンバーシップ度を持つ事例がたくさんあるのに対し、ベクトル空間の別の端は弱いメンバーシップ度の事例しかないかもしれない。第3の柱は、それぞれの原因条件の組み合わせについての、当該原因条件の組み合わせが結果の部分集合であるという議論の証拠の整合性の評価である。真理表アルゴリズムは、これら3つの柱を利用してクリスプ集合の真理表を構築するもので、その点ではクリスプ集合でのアルゴリズムと似通っている。この節では、複数のファジィ集合分析の結果をクリスプ集合の真理表に記録し、その表を分析するといったステップを説明する。

[7]（訳注）　例えば、原因条件が 3 つある場合、原因条件の全ての組み合わせは、1 辺の長さが 1 の立方体の各頂点によって表される。この各頂点のことをここでは端（corner）と呼んでいる。

データ

ファジィ集合のデータは、他のプログラムからインポートすることもできるし、第1章と第2章で説明したようにfs/QCAで作成することもできる。この章では、弱い階級投票（class voting）がある国々という、Ragin(2008)で用いた例を使用する。下の表は、データシートを表している。



Case	name	gdpf	inqf	manf	unf	weakcv
1	AUS	0.9	0.7	0.3	0.7	0.7
2	BEL	0.7	0.1	0.1	0.9	0.7
3	DEN	0.7	0.3	0.1	0.9	0.1
4	FRA	0.7	0.9	0.1	0.1	0.9
5	GER	0.7	0.9	0.3	0.3	0.7
6	IRE	0.1	0.7	0.9	0.7	0.9
7	ITA	0.3	0.9	0.1	0.7	0.7
8	NET	0.7	0.3	0.1	0.3	0.9
9	NOR	0.7	0.3	0.7	0.9	0.1
10	SWE	0.9	0.3	0.9	1	0
11	UKN	0.7	0.7	0.9	0.7	0.3
12	USA	1	0.9	0.3	0.1	1

name	国の略称
gdpf	豊かさ
inqf	重大な賃金格差
manf	強い製造部門
unf	強い組合
weakcv	弱い階級投票

間隔尺度や比率尺度は、マニュアルの第2章（データエディタ）やRagin(2008)で説明されているような「キャリブレーション」の手続きを使って、ファジィ集合のメンバーシップ度に変換できる。

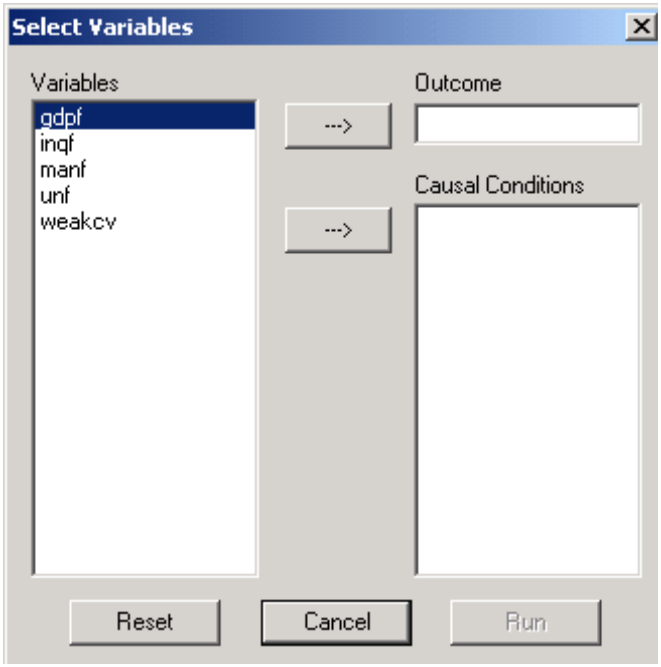
分析

真理表アルゴリズムは、2つのステップの分析手続きからなっている。第1ステップでは、ファジィ集合のデータから真理表を作成する。これは、それぞれの条件の組み合わせに対する結果を指定し、どの条件の組み合わせを分析に含めるべきかを決定する、といったことを含む。第2ステ

ップでは、簡単化する原因条件と結果を特定する。これらのステップは、互いに関連を持って行われなければならず、それぞれの分析の際には両方のステップを実行しなければならない。

- 出力ウィンドウにデータをリスト化するには、次のように選択する。
  Analyze（分析）
    Fuzzy Sets（ファジィ集合）
      Truth Table Algorithm（真理表アルゴリズム）

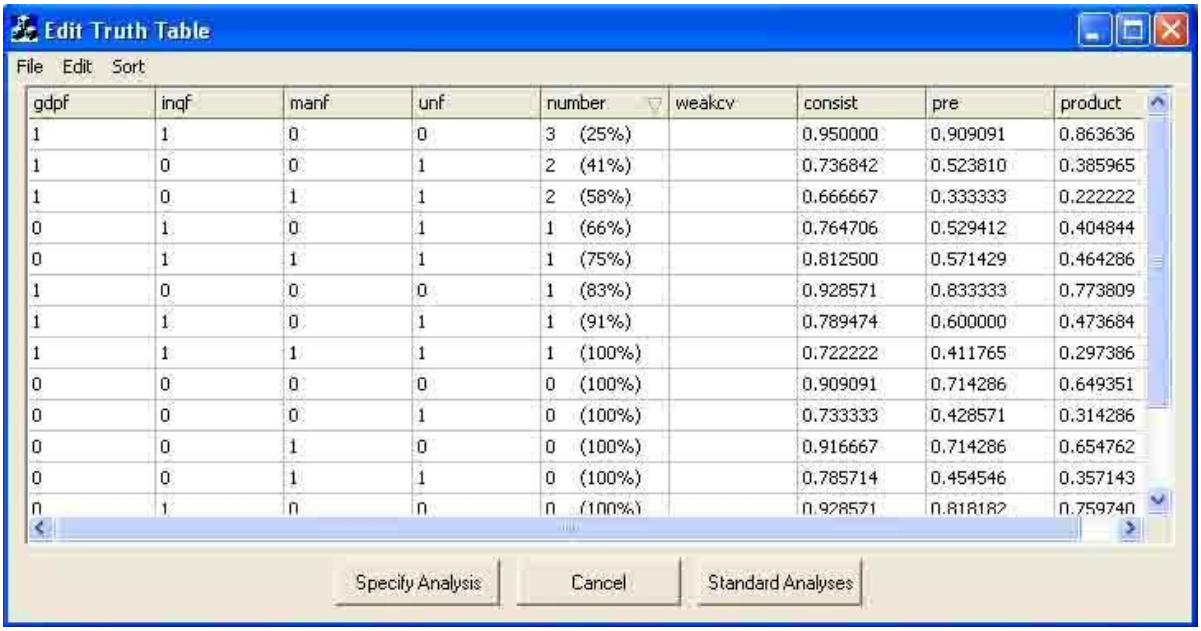
次のようなウィンドウが開く:



- 説明したい変数を特定し、それをOutcome（結果）のフィールドに移す。

- 原因条件を1つずつ選んで、Causal Conditions（原因条件）のフィールドに移していく。

- Run（実行）のボタンをクリックすると、次のようなウィンドウが現れ、真理表が表示される。



- 真理表は、2^k個（kは原因条件の数を表す）の行を持ち、これらの行は原因条件の全ての可能な組み合わせを表している。1と0という数字は、ファジィ集合の原因条件によって定義されるベクトル空間の様々な端を表している。それぞれの行について、以下のような変数の値が作成される。

number　ベクトル空間の当該端において、0.5よりも大きいメンバーシップ度を持つ事例の数。かっこの中に示されているのは、事例の累積的なパーセンテージである。ベクトル空間の端は、numberの数が大きい方から順に、上から並べられている。

consist　ベクトル空間の当該端が、結果の整合的な部分集合になっている度合い[8]（クリスプ集合では、これは真理表のそれぞれの行における、当該結果が生じる事例の割合である）。

pre　誤差減少率（PRE）に準ずる計算に基づいた、整合度の別の測定方法[9]（クリスプ集合の分析では、consistと等しい）。

product　consistとpreの積。これは、真理表の上の方の行における整合度のギャップを特定し、結果のための整合度の閾値を確立するのに役立つ[10]。

---

[8]（訳注） 注 3 と同様、consist は consistency（整合度）の略。現在のバージョン（2009 年）では、raw consistency（粗整合度）と呼ばれている。

[9]（訳注） 注 4 と同様、pre は proportional reduction in error（誤差減少率）の略。現在のバージョン（2009 年）では、PRI consistency（PRI 整合度）と呼ばれている。PRI は proportional reduction in inconsistency（不整合減少率）の略。具体的な計算方法は、95 ページで説明する。

[10]（訳注） 次ページで説明されているように、真理表は整合度の値の大きいものが上になるようにソートされる。そして次に、整合度の閾値の値を設定するが、その際にある行とその行のすぐ下の行の整合度を比べた場合に整合度に大きな差（ギャップ）があるところを見つけ出し、それをもとに閾値を設定する。ファジィ集合の場合、この作業に consist だけでなく product の値も使用できるということをここでは述べている。

結果としてラベルされている列（今の例ではweakcv)は空白になっていることに注意する。それぞれの条件の組み合わせについて、以下で説明する手続きを用いて結果を決定しスプレッドシートに入力することは、研究者の仕事である。

- 研究者はまず、原因条件で定義されたベクトル空間のそれぞれの領域に存在する事例の数に基づいて、それぞれの条件の組み合わせ（ベクトル空間の端）を、関連のある組み合わせと関連のない組み合わせに分類するルールを構築しなければならない。そのためには、それぞれの条件の組み合わせで0.5よりも大きいメンバーシップ度を持つ事例の数に基づいて、度数の閾値を選択すればよい。これはnumberの列に表示されている。分析する事例の総数が比較的少ないときは、度数の閾値は1か2にすべきである。しかし総数が大きいときは、より大きな閾値が使われるべきである。非常に重要なのは、原因条件の組み合わせにおける事例の分布を調べ、ベクトル空間の中で最も事例の多い領域を見つけ出すことである。一般的に、選択された条件の組み合わせで、事例の少なくとも75-80%が捕捉されているべきである。

- numberの列の任意のセルをクリックして次のように選択するることで、事例を度数でソートできる。

  Sort（ソート）

  Descending（降順）

- ソートして度数の閾値を選択した後、閾値を満たさない行を全て削除する。事例がnumberで降順にソートされている場合、閾値よりも下の最初の事例をクリックして、次のように選択すればよい。

  Edit（編集）

  Delete current row to last row（現在の行から最後の行までを削除）

  事例がソートされていない場合は、閾値を満たさない事例の行をクリックして次のように選択することで、それらを個々に削除することができる。

  Edit（編集）

  Delete current row（現在の行を削除）

- 次のステップは、結果の整合的な部分集合となる条件の組み合わせと、そうでないものを区別することである。これを決定するには、consist, pre, productの列に表示される集合論的な整合性の測定値を使用すればよい。consistの列の値が0.75以下の場合は、非常に整合性がないことを表す。整合度の分布を評価するために、整合度を降順にソートするのが有用である（これは度数の閾値を満たさない行を削除した後に行うべきである)。ソートするには、consist, pre, productのどれかの列の任意のセルをクリックして、次のように選択すればよい。

  Sort（ソート）

Descending（降順）

上の方の整合度を見て、整合度の閾値を定立するのに使えるようなギャップを見つける。複数の閾値を検討したり、閾値の値を上げ下げして結果を評価したりすることが常に可能なことを頭に置いておくこと。

- 次に、どの条件の組み合わせで結果が生じており、どの条件の組み合わせでは結果が生じていないものとするかを指定しなければならない。それぞれの条件の組み合わせのうちで、整合度の値が閾値以上となるものの結果の列（今の例ではweakcv）に1を入力する。それぞれの条件の組み合わせのうちで、整合度の値が閾値を満たさないものの結果の列に0を入力する。

- 別の方法として、Delete and code（削除とコーディング）の機能を使うと、今のプロセスを自動化できる。以下のように選択する。

  Edit（編集）

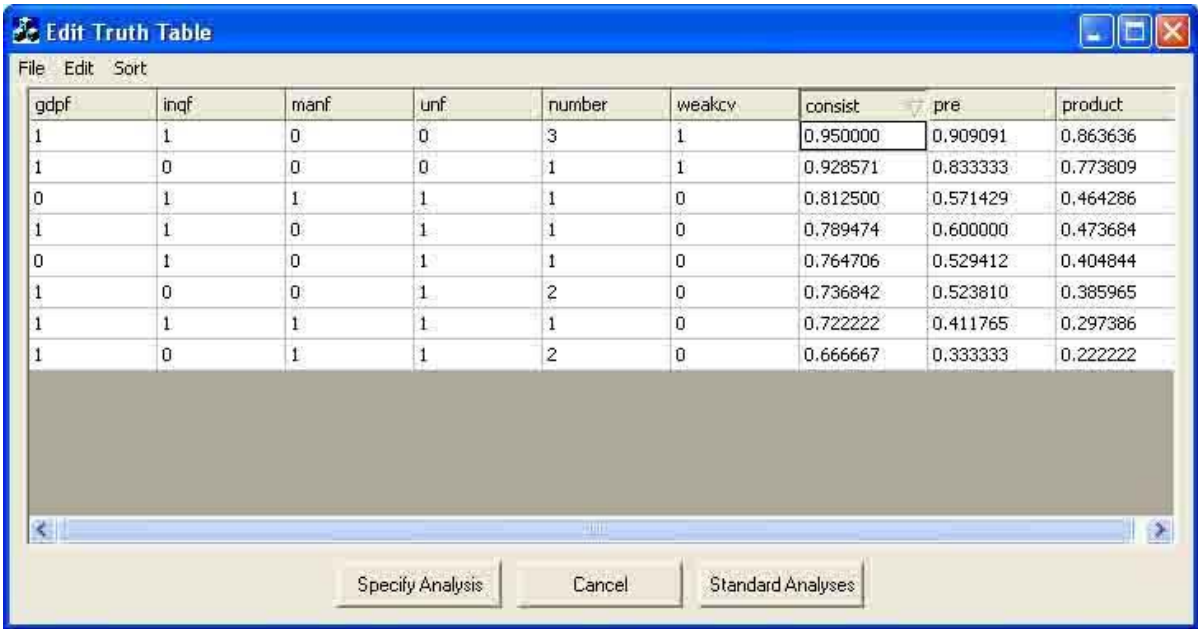
  Delete and code（削除とコーディング）

1番目のフィールドでは、度数の閾値を選択する。デフォルトの数字は1となっているが、自分の選択した閾値をタイプすることで閾値を変えることができる。2番目のフィールドでは、整合度の閾値を選択する。デフォルトの数字は0.8となっているが、自分の選択した閾値をタイプすることで閾値を変えることができる。

OKをクリックする。度数の閾値を満たさない行をプログラムが削除し、選択した整合度の閾値に応じて結果の列に0か1がコーディングされる。

- 以下のウィンドウは、次の2つの作業後の真理表を表している。

1. 度数の閾値として1を適用し、観察されなかった条件の組み合わせ（8つ）を排除する。
2. 整合度の閾値として0.9を適用し、0.9以上の整合度を持つ条件の組み合わせ（2つ）のweakcvの列には1を入力し、0.9より小さい整合度を持つ条件の組み合わせ（6つ）のweakcvの列には0を入力する。

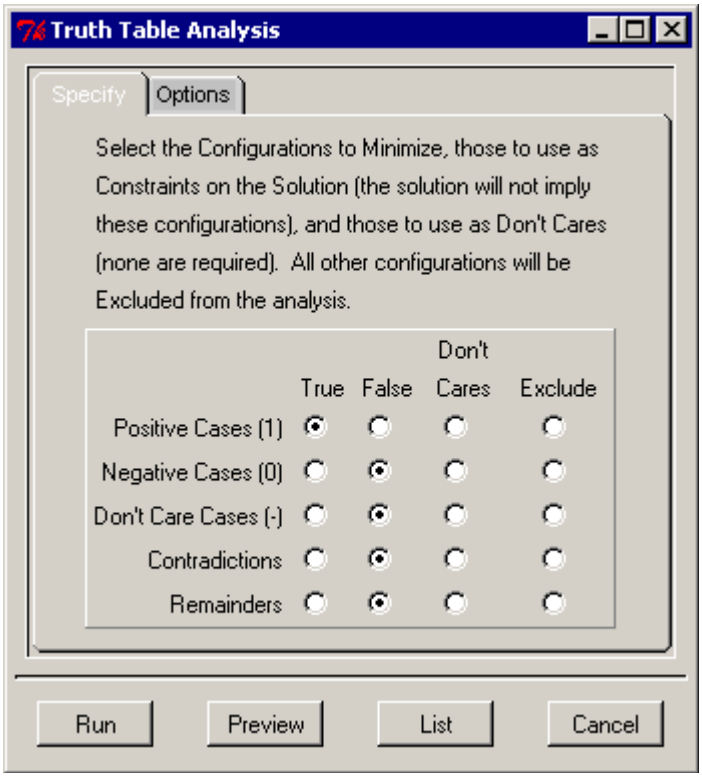


ここから、Specify Analysis（分析を指定）を選択するか、Standard Analyses（標準分析）を選択するか、という2つの分析の可能性がある。標準分析の選択が推奨される。なぜなら、intermediate solution（中間解）を導けるのはこちらだけだからである。

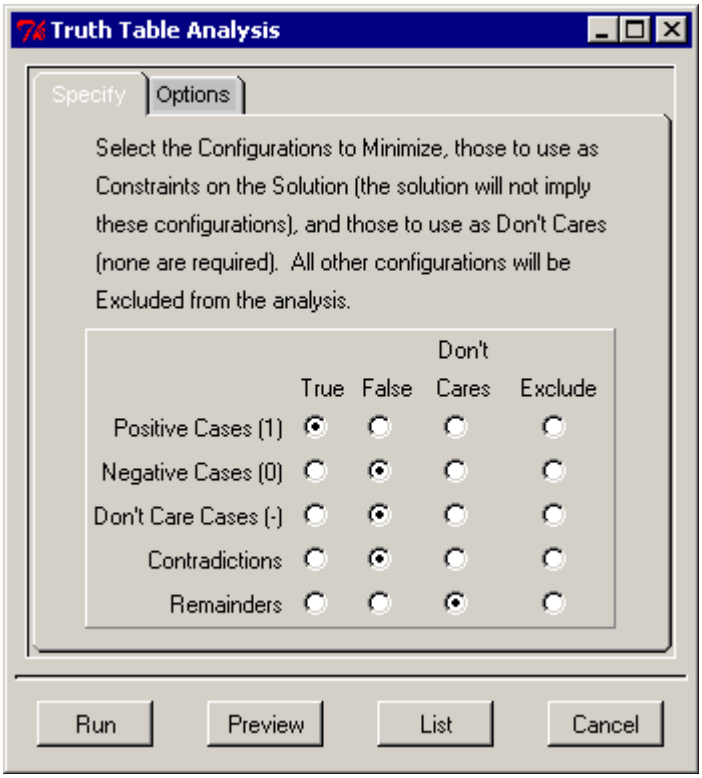
分析指定オプション

- 真理表を作成した後、Specify Analysis（分析を指定）を選択すると、Truth Table Analysis（真理表分析）ウィンドウが現れる。

- Specify（指定）パネルにおいて、Positive Cases（結果の値が1の事例）でTrueを選択し、他は全てFalseを選択すると、最も複雑な解が導かれる。この場合、ウィンドウは次のようになる。



- 最も簡略な解を導くには、Positive Cases（結果の値が1の事例）でTrueを選択し、Negative Cases（結果の値が0の事例）、Don't Care Cases（結果の値が-の事例）、Contradictions（矛盾した事例）でFalseを選択し、Remainders（残余部分）でDon't Cares（ドント・ケア）を選択する。この場合、ウィンドウは次のようになる。



- 他のオプション（主項や度数の出力に関するもの）は全て、クリスプ集合のところで前述したのと同様の手続きにしたがって設定することができる。ただし、一般的に言って、真理表アルゴリズムを使用する際にはこれらのオプションは役に立たない。

標準分析オプション

真理表ができあがったら、Standard Analyses（標準分析）を選択する。標準分析では、complex（複雑）、parsimonious（簡略）、intermediate（中間）の解が得られる。Standard Analyses（標準分析）はSpecify Analysis（分析を指定）よりもお薦めである。

クリスプ集合でのStandard Analyses（標準分析）手続きの議論を参照しなおしていただきたい。ファジィ集合の手続きは、それとパラレルになっている。complex（複雑）、parsimonious（簡略）、intermediate（中間）という3つの解が導かれる。3つの解は、残余部分の組み合わせの扱いの違いから生まれる。

Complex（複雑）：remainders（残余部分）は全てfalseに設定される。反事実はなし。

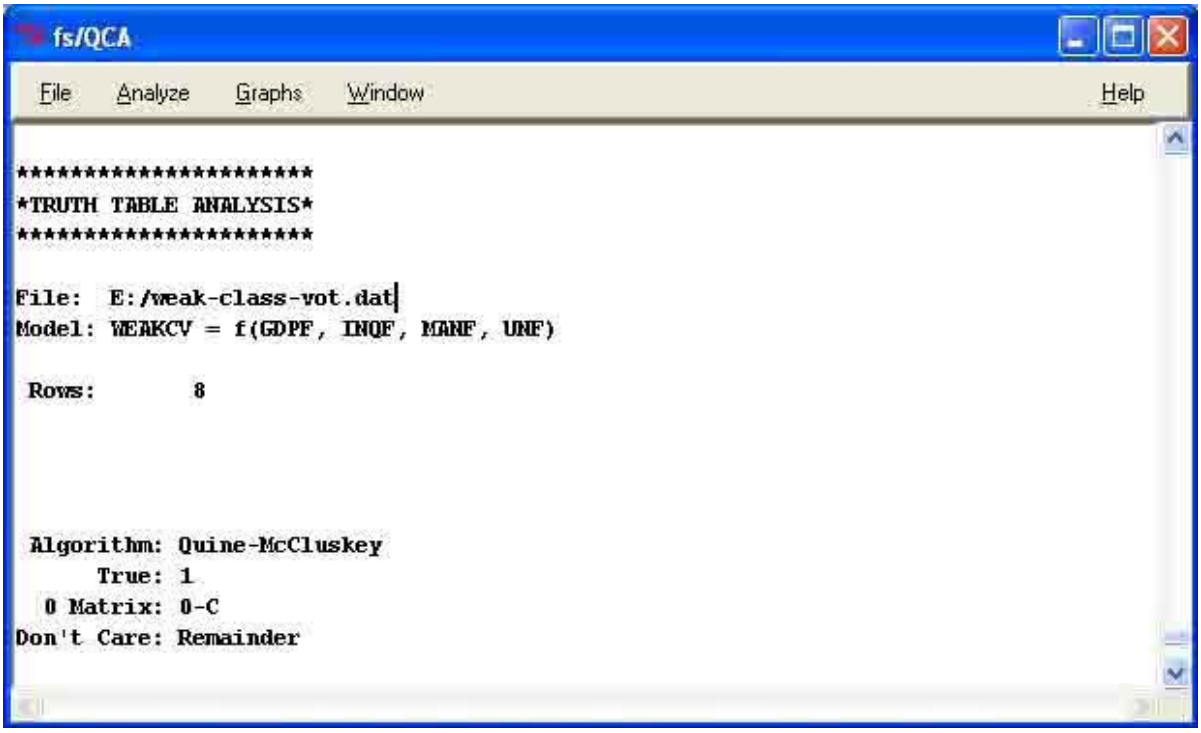
Parsimonious（簡略）：「容易な」反事実か「困難な」反事実かに関わらず、論理的に単純な解を導くのに役立つ残余部分は全て使用される。

Intermediate（中間）：「容易な」反事実の残余部分のみが解に組み込まれる。「容易」か「困難」かは、それぞれの原因条件と結果の間のつながりに関してユーザーからもたらされた情報に基づいて指定される。

D）分析指定オプションの出力

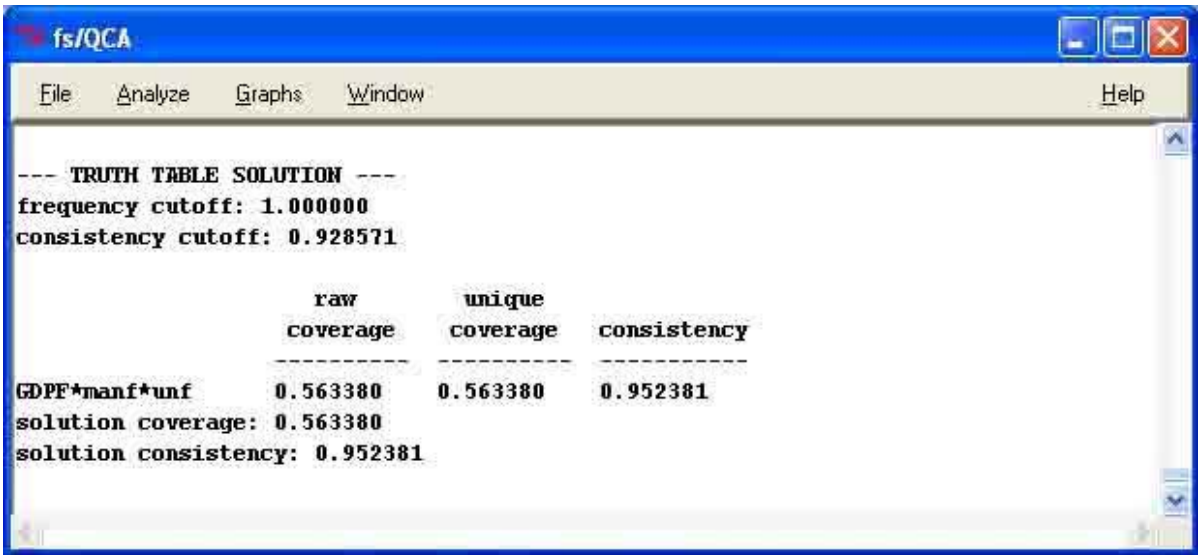
真理表を簡単化すると、メインウィンドウに以下のような出力が表示される。表示されている出力は、上で説明したSpecify Analysis（分析を指定）オプションを使用して得られたcomplex solution（複雑解）である。

出力の最初の部分にはデータが表される。最初の2つの行は、ファイルのディレクトリと指定したモデルを表している。次の行では、分析のために読み込んだ真理表の行（スプレッドシートにコーディングされたもの）の数をリスト化している。その後に、使用したアルゴリズムの種類、簡単化した条件の組み合わせ、設定した制約が表示される。下の例では、結果が生じる（値が1）条件の組み合わせが簡単化されており、結果が生じない（0）、ドント・ケア（-）、矛盾（C）の条件の組み合わせは制約（false）に設定され、残余部分（L）はドント・ケアに設定されている。



出力の最後の部分では、分析の解が表示される。最初に、度数と閾値の区切り値がリスト化される。その次が解である。Specify Analysis（分析を指定）オプションを選択した場合、Truth Table Solution（真理表解）というセクションが1つ表示される。Standard Analyses（標準分析）を選択した場合、3つの解（複雑 complex、簡略 parsimonious、中間 intermediate）が表示される。

この例では、最も複雑な解として次のような解が表示される。



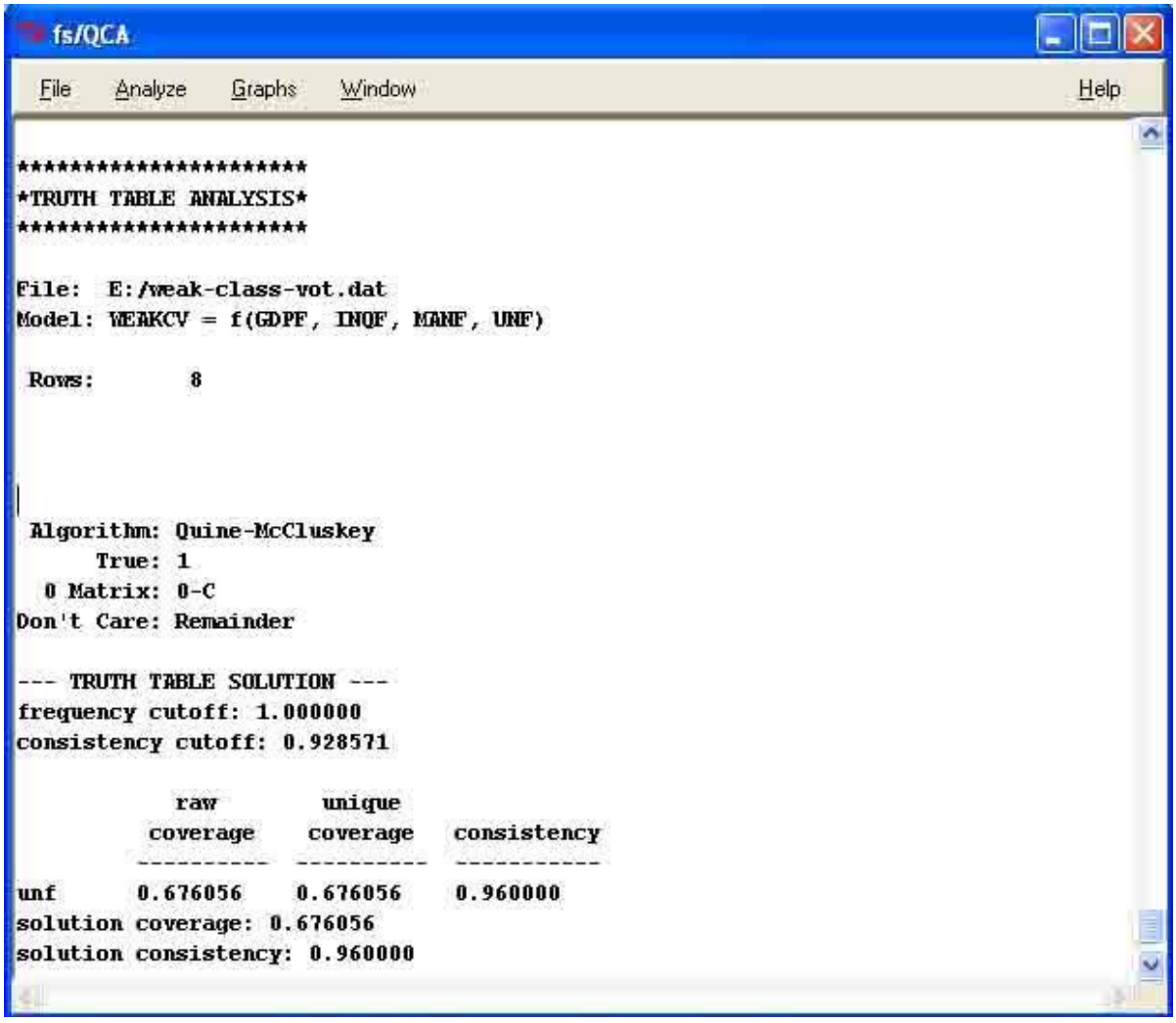
弱い階級投票は、裕福な国の集合での高いメンバーシップ度、製造業の労働者の割合の高い国の集合での低いメンバーシップ度、労働組合が強い国の集合での弱いメンバーシップ度の論理積

である。

最初の2つの行は、度数と閾値の区切り値をリスト化している。閾値の区切り値は、ユーザーが指定した閾値よりも高い真理表中の整合度の中で、最小のものがリストされる。ここでは、0.9が整合度の閾値として設定され、0.9よりも大きい実際の最小の値は0.928571であった。

解では、1行ずつそれぞれ別々の因果経路が表示される（今の例では、経路はGDPF*manf*unfというただ1つである）。出力では、解のそれぞれの項や解全体についての整合度（consistency）と被覆度（coverage）も計算される（これらの計算については後ほど説明する）。

今の例の真理表の最も簡略な解についての出力は次のようになる。

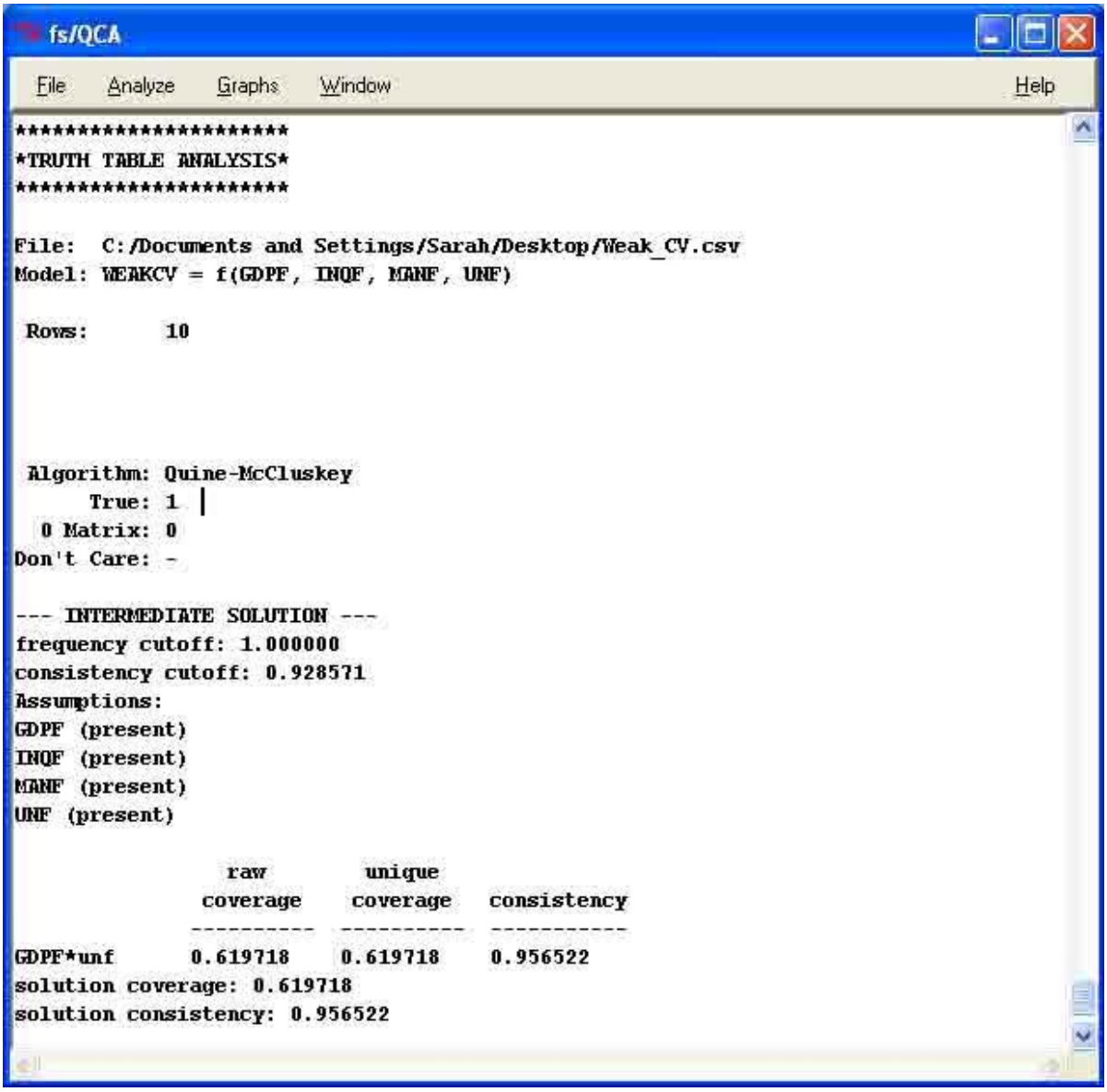


この場合も出力では、使用したアルゴリズムの種類、簡単化した条件の組み合わせ、設定した制約が表示される。今の例では、結果が生じる（値が1）条件の組み合わせが簡単化されており、結果が欠如している（0）、ドント・ケア（-）、矛盾（C）の条件の組み合わせは制約（false）に設定され、残余部分（R）はドント・ケアに設定されている。

解では、弱い階級投票の因果経路がただ1つ示されている。労働組合が強い国の集合でのメンバーシップ度が低い国では、弱い階級投票が生じる。

E）標準分析オプションの出力

Standard Analyses（標準分析）の出力は、次のように若干異なっている。



最も顕著な違いは、複雑解、中間解、簡略解が全て表示されることである。上の図で表示されているのは中間解である。Assumptions（仮定）という出力の部分では、Intermediate Solution（中間解）ウィンドウで選択したオプションが表示される。今の例では、それぞれの原因条件が存在する場合に、結果の発生に影響を与えるというように選択をした。

この解では、弱い階級投票は、裕福な国の集合での高いメンバーシップ度と、労働組合が強い

国の集合での弱いメンバーシップ度の論理積である。

F）整合度と被覆度

出力には、解のそれぞれの項と解全体についての整合度と被覆度の値が含まれる。整合度（consistency）は、解の項や解全体が結果の部分集合になっている度合いを表している。被覆度（coverage）は、解のそれぞれの項や解全体によって結果がどれぐらいカバー（あるいは説明）されているかを表している。これらの値は、もとのファジィ集合のデータセットを、（1つ以上の項から成る）解を考慮しながら検討することにより計算される。より具体的には、次のようなデータ表を考えてみる。この表には、3つの原因条件（A, B, C）と結果（Y）があり、全てファジィ集合で値が記されている。

原因条件のメンバーシップ度			結果のメンバーシップ度	解のメンバーシップ度			解の整合度		
A	B	C	Y	A*B	A*C	A*B+A*C	C_{A*B}	C_{A*C}	$C_{A*B+A*C}$
0.8	0.9	0.8	0.9	0.8	0.8	0.8	0.8	0.8	0.8
0.6	0.7	0.4	0.8	0.6	0.4	0.6	0.6	0.4	0.6
0.6	0.7	0.2	0.7	0.6	0.2	0.6	0.6	0.2	0.6
0.6	0.6	0.3	0.7	0.6	0.3	0.6	0.6	0.3	0.6
0.8	0.3	0.7	0.8	0.3	0.7	0.7	0.3	0.7	0.7
0.6	0.1	0.7	0.9	0.1	0.6	0.6	0.1	0.6	0.6
0.7	0.4	0.2	0.3	0.4	0.2	0.4	0.3	0.2	0.3
0.2	0.9	0.9	0.1	0.2	0.2	0.2	0.1	0.1	0.1
0.1	.6	0.2	0.2	0.1	0.1	0.1	0.1	0.1	0.1
0.2	0.1	0.7	0.3	0.1	0.2	0.2	0.1	0.2	0.2
0.3	0.1	0.3	0.3	0.1	0.3	0.3	0.1	0.3	0.3
0.1	0.2	0.3	0.2	0.1	0.1	0.1	0.1	0.1	0.1
合計			6.2	4.0	4.1	5.2	3.8	4.0	5.0

この分析についての関連した出力は下に示されている。解は、A*B + A*Cという2つの項から成る。整合度と被覆度を計算するには、いくつかの媒介となる値を最初に計算する必要がある。結果のメンバーシップ度（Σ_Y）は、データの全ての事例についての、結果の集合におけるメンバーシップ度の和である。解のそれぞれの項における事例のメンバーシップ度は、当該項のそれぞれの原因条件における事例のメンバーシップ度の最小値である。1番目の項のメンバーシップ度（Σ_{A*B}）は、解の1番目の項の集合における全ての事例のメンバーシップ度を足したものである。同様に、2番目の項のメンバーシップ度（Σ_{A*C}）は、解の2番目の項の集合における全ての事例のメンバーシップ度を足したものである。解のメンバーシップ度（$\Sigma_{A*B+A*C}$）は、全ての事例について解の項のメンバーシップ度の最大値をとって、それを足したものとして定義される。

```
            raw         unique
          coverage    coverage    consistency
         ----------  ----------  -----------
A*B+     0.612903    0.161290    0.950000
A*C      0.645161    0.193548    0.975610
solution coverage:   0.806452
solution consistency: 0.961538
```

整合度（consistency）は、解のそれぞれの項が結果の部分集合になっている度合いである。整合度を計算するには、まずそれぞれの事例の整合度を計算する。解のどの項についても、その項におけるある事例のメンバーシップ度が結果でのメンバーシップ度以下であるならば、その事例は整合的である[11]。解の項における事例のメンバーシップ度が結果におけるメンバーシップ度よりも大きい（すなわち不整合的である）ならば、当該事例には結果におけるメンバーシップ度と等しい値が与えられる。これらの値を足し合わせて（ΣC_{A*B}）、解の項におけるメンバーシップ度の合計（Σ_{A*B}）で割る。以上から、解の1番目の項の整合度は$\Sigma C_{A*B} / \Sigma_{A*B} = 3.8/4 = 0.95$であり、2番目の項の整合度は$4.0/4.1 = 0.976$である。

解整合度（solution consistency）は、解（解の項の集合）が結果の部分集合になっている度合いである。それぞれの事例の、解の項（A*B + A*C）のメンバーシップ度の中の最大値が、結果におけるメンバーシップ度と比較される。解のメンバーシップ度が結果のメンバーシップ度以下であるならば、当該事例には解のメンバーシップ度と等しい値が与えられる。解のメンバーシップ度が結果のメンバーシップ度よりも大きい（すなわち不整合的である）ならば、当該事例には結果のメンバーシップ度と等しい値が与えられる（すなわち、2つの値のうち小さい方が与えられる。）これらの値を足し合わせて、解のメンバーシップ度の合計で割る（$\Sigma C_{(A*B+A*C)} / \Sigma_{(A*B+A*C)}$）。今の例における解整合度は、5.0/5.2=0.962である。

解被覆度（solution coverage）は、解全体で説明される結果の割合である。全ての事例の整合度を足し合わせて、結果のメンバーシップ度の合計で割る。今の例では、$\Sigma C_{(A*B+A*C)} / \Sigma_Y = 5.2/6.2 = 0.806$となる。

粗被覆度（raw coverage）は、解のそれぞれの項で説明される結果の割合である。解のそれぞれの項の粗被覆度は、解の当該項の整合度の合計を結果のメンバーシップ度の合計で割ることで、もとのデータから計算される。解の1番目の項の粗被覆度は$\Sigma C_{A*B} / \Sigma_Y = 3.8/6.2 = 0.613$であり、2番目の項の粗被覆度は$4.0/6.2 = 4.0/6.2 = 0.645$である。

---

[11] （訳注）この場合には、当該事例には解の項でのメンバーシップ度と等しい値が与えられる。すなわち、当該事例の解の項におけるメンバーシップ度と結果におけるメンバーシップ度が比較され、小さい方が選ばれるということである。

固有被覆度（unique coverage）は、解の個々の項単独で説明される（他の項でカバーされない）結果の割合である。これを計算するには、まず解から当該項を取り除いて解被覆度を計算する。今の例では、解の1番目の項（ΣC_{A*B}）を取り除いた後の解の被覆度は、単純に$\Sigma C_{A*C} / \Sigma Y$である（解にn個の項がある場合、取り除いた後の解はn-1個の項を含む）。もとの解被覆度から、当該項を取り除いた後の解被覆度を引けば、取り除いた項の固有被覆度が与えられる。解の1番目の項（ΣC_{A*B}）の固有被覆度は、$(\Sigma C_{(A*B+A*C)} / \Sigma Y) - (\Sigma C_{A*C} / \Sigma Y) = (5.0/6.2) - (4.0/6.2) = 0.161$である。2番目の項の固有被覆度は、$(5.0/6.2) - (3.8/6.2) = 0.194$である。

【訳注】真理表におけるpre（新しいバージョンではPRI整合度）の意味について（このマニュアルの52ページと84ページを参照）

※以下の内容は、Charles Ragin教授にEメールでご教示いただいた。

ファジィ集合のQCAにおいて、真理表分析の整合度とは、原因が結果の部分集合になっている度合いのことであった。公式としては、次のように表される（このマニュアルの第5章のFやRagin(2008:44-68）参照）。Xを原因条件の集合、Yを結果の集合とする。それぞれの集合の要素をx_i, y_iとする。このとき整合度は、

$$\frac{\Sigma \min(x_i, y_i)}{\Sigma x_i}$$

となる。それに対し、pre（新しいバージョンではPRI整合度）は次のように計算する。

$$\frac{\Sigma \min(x_i, y_i) - \Sigma \min(x_i, y_i, \sim y_i)}{\Sigma x_i - \Sigma \min(x_i, y_i, \sim y_i)}$$

これは、以下のことを考慮した指標である。ファジィ集合では、x_iがy_iよりも小さくかつ~y_iよりも小さい場合が生じる。例えば、x_1=0.4, y_1=0.5とすると、~y_1=1-0.5=0.5なので、x_1はy_1よりも小さくかつ~y_1よりも小さい。このとき、x_iがy_iの部分集合でも~y_iの部分集合でもあるということになる。このように肯定も否定も成り立ってしまう場合は、整合度の計算であまり重要性を持たないと思われるため、この部分を計算から除くことを考えたのがこの指標である。この指標の場合、分母と分子の両方から、XとYと~Yの共通部分を除くという計算方法をとっている。

なぜこれをpre（新しいバージョンではPRI整合度）と呼ぶかであるが、計算方法が統計学におけるPRE（proportional reduction in error, 誤差減少率）の指標に似ているからである。誤差減少率とは、次のような形の式となる指標の総称である（太郎丸(2005:76)などを参照）。

$$\frac{\text{情報なしでの誤差 - 情報ありでの誤差}}{\text{情報なしでの誤差}}$$

この式は、あるデータについて、何らかの情報が付加された場合に、その情報をもとにするとどれだけ誤差を減少させることができるかを表すものである。ガットマンのラムダ（Guttman's lambda）、グッドマンとクラスカルのタウ（Goodman and Kruskal's tau）、回帰分析の決定係数（R^2）などがこれにあたる。例えば、回帰分析の決定係数は次のようにPREの構造になっている。

$$\frac{\text{全変動 - 残差変動}}{\text{全変動}}$$

pre（PRI整合度）も、実はこれと似たような構造になっている。上のpre（PRI整合度）の公式を変形すると、次のようになる。

$$\frac{[\Sigma x_i - \Sigma \min(x_i, y_i, \sim y_i)] - [\Sigma x_i - \Sigma \min(x_i, y_i)]}{\Sigma x_i - \Sigma \min(x_i, y_i, \sim y_i)}$$

この式は、次のような構造になっている。

$$\frac{\text{Y でも~Y でも不整合な部分 - Y で不整合な部分}}{\text{Y でも~Y でも不整合な部分}}$$

すなわち、XがYの部分集合か~Yの部分集合かどちらになるか分からない状態と、XがYの部分集合であるとした状態を比較して、不整合性がどれだけ減るかを計算していることになる。そこで古いバージョンではそのままpreと呼ばれ、新しいバージョンではPRI（proportional reduction of inconsistency, 不整合減少率）と呼ばれているのである。

参考文献

Mendelson, E. (1970) *Boolean Algebra and Switching Circuits*, McGraw-Hill（大矢建正訳『ブール代数とスイッチ回路』マグロウヒル好学社、1982年）.

McDermott, R. (1985) *Computer-Aided Logic Design,* Howard W. Sams.

Ragin, C. C. (1987) *The Comparative Method: Moving beyond Qualitative and Quantitative Strategies*, University of California Press (鹿又伸夫監訳『社会科学における比較研究―質的分析と計量的分析の統合にむけて』ミネルヴァ書房、1993年).

Ragin, C. C. (2000) *Fuzzy-Set Social Science*, University of Chicago Press.

Ragin, C. C. (2008) *Redesigning Social Inquiry: Fuzzy Sets and Beyond*, University of

Chicago Press.
Rihoux, B. and C. C. Ragin (2008) *Configurational Comparative Methods: Qualitative Comparative Analysis (QCA) and Related Techniques*, Sage Publications, Inc.
Roth, C. (1975) *Fundamentals of Logic Design*, West.
Weber, M.(1949) *The Methodology of the Social Sciences*, Free Press.（富永祐治・立野保男訳『社会科学方法論』岩波書店、1936 年）.
太郎丸博（2005）人文・社会科学のためのカテゴリカル・データ解析入門、ナカニシヤ出版.