RB-003

ラダーダイアグラムの検索的解析環境の構築

Key Word Combinable Program Analysis Environment for Ladder Diagram

仲井 勘†
Satoru Nakai

1．はじめに

産業用オートメーションシステムには，シーケンサ（Programmable Logic Controller）が多く用いられる．シーケンサは，センサやスイッチ等の入力機器の ON/OFF 条件に応じてバルブやランプ等の出力機器を ON/OFF することで，シーケンス制御を実現する専用コントローラである．シーケンサが実行する制御プログラムの記述には，ラダーダイアグラム（Ladder Diagram：以下，ラダーと略す）が最もよく使われている．

近年，シーケンサの性能向上や制御内容の高度化に伴い，ラダーが大規模・複雑化している．一方で，過去のラダー資産を流用する開発が継続して行われている．その結果，不具合原因の特定やその修正または機能追加による影響範囲の把握といった作業が煩雑化している．

ラダーの開発を支援するエンジニアリングツールには，様々な目的・用途に幅広く利用できる基礎的なプログラム解析機能が備わっている．ただし，１つの基礎的な機能だけでは直ちに所望の解析結果を抽出できないため，いくつかの解析結果を別途手動で組み合わせ，目的・用途に応じた所望の解析結果に仕立てる必要がある．これに対して，特定の目的を達成するためのラダー解析技術の研究例[6]もあるが，利用場面が限定されるため，一般ユーザが利用できるツールとしての実現には至っていない．すなわち，現状のツールが備える解析機能では，ラダー開発に係る作業の煩雑化を解決するには十分ではないといえる．

そこで本研究では，従来の基礎的なラダー解析機能を整理し正規化することで，高度な解析を基礎的な機能の組み合わせで実現できる機能体系を確立し，この機能体系に基づいて解析条件の自由な組み合わせによる絞り込みが可能な検索的解析支援ツールを開発した．本ツールは，解析条件を条件式の形式で入力し，解析結果に対してさらに別の解析条件を適用したり，解析結果どうしの論理演算を行うといったデータベース検索のような絞り込みが可能であり，本稿ではこれを検索的解析環境と呼ぶことにする．本ツールの開発により，制御内容に直接関与する解析結果を効率的に抽出できるようになり，旧来より稼動する多くのラダーに対する不具合改修や機能追加が容易となる．

本稿の構成は次のようになっている．先ず準備としてラダーの特徴を 2.で説明し，従来のラダー解析機能およびその課題について 3.で述べる．次に参考技術としてデータベース検索について 4.で確認する．そして 5.でラダー解析機能の体系化について述べ，これに基づいて開発した検索的解析支援ツールと解析例を 6.で紹介する．7.では，プログラム解析に関する他の研究について本研究との関連を考察する．最後に 8.で本稿をまとめる．

†三菱電機（株）先端技術総合研究所

2．ラダーダイアグラムの特徴

シーケンス制御は元来，リレーにより実現・構成されていた．ラダーは，リレーによる論理回路を記述するために考案されたものである．この歴史的な理由から，シーケンサの制御プログラムにも，ラダーが継続的に使われている．

シーケンサでは，接続する入力機器の状態を入力変数に格納し，出力機器の状態は出力変数を通じて制御する．制御内容の途中結果をシーケンサの内部変数に保持することもできる．これら変数を用いてラダーを記述する．

当社のシーケンサでは，入力変数を X デバイス，出力変数を Y デバイス，内部変数を M デバイスと呼ぶ．例えば，入力スイッチ X1 と X2 が両方とも ON した条件で内部状態 M10 が ON するラダーは図 1 のようになる．また，内部状態 M10 と M20 のいずれか一方が ON した条件で出力ランプ Y3 が ON するラダーは図 2 のようになる．

このようなラダーがまとまって 1 つの MAIN プログラムとなる．MAIN プログラムは上のラダーから順次実行され，その制御結果を受けて下のラダーが動作し，プログラムの終わりまでくると，はじめに戻って実行を繰り返す．プログラム実行の一巡はスキャン処理と呼ばれ，それに要する時間はスキャンタイムと呼ばれる．

上述の歴史的な理由により，ラダー記述に用いる変数はグローバル変数であり，全ての制御内容を 1 つの MAIN プログラムに記述したラダーが現在でも数多く稼動している．

最近では，サブルーチンや FB(Function Block)といったプログラムモジュール（POU：Program Organization Unit）によるラダーの部品化も浸透し始めている（図３参照）．

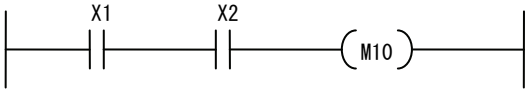

図 1：AND 条件ラダー

図 2：OR 条件ラダー

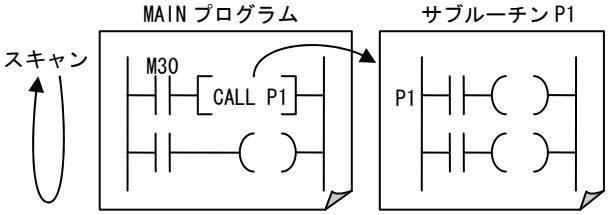

図 3：サブルーチンを使ったラダー

ラダーのサブルーチンは，C 言語などの高級言語における GOTO 文でジャンプした先の処理や，引数を伴わない関数に相当する．ラダーでは，サブルーチンを同じプログラムファイル内に記述しても，個別のプログラムファイルとして記述してもどちらでもよい．FB は，C 言語などの引数を伴う関数や C++言語などのオブジェクトのメソッドに相当する．FB は個別のプログラムファイルとして記述する．

このような特徴を持つ最近のラダーの開発では，スキャンタイムを短くして制御の応答性を高めるため，条件が成立した時のみサブルーチンコールするプログラム構造にして，通常のスキャンで実行されるラダーの数を減らす等の工夫がなされる．図 3 の例では M30 が ON の時のみサブルーチン P1 が実行される．

3. 従来のラダー解析機能および課題

シーケンサのラダープログラムでは，数多くの制御状態をグローバルな内部変数で保持する．このため，１つの変数の値の変化に影響される他の変数の数が非常に多いプログラム構造となる．このような特徴を持つラダーの解析を支援するために，シーケンサのエンジニアリングツールには次のような機能が備わっている（図 4 参照）；

①POU 間の呼び合い関係の抽出
②プログラム中で使用している変数の一覧抽出
③プログラム中の変数間の依存関係の抽出

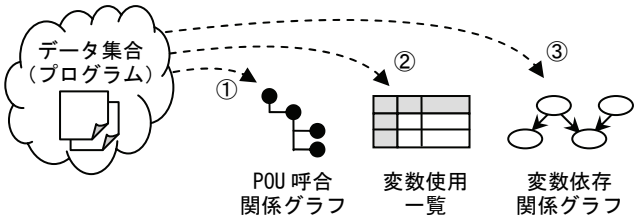

図 4：従来のラダー解析機能

サブルーチンコールが使われているラダープログラムに対しては，①を利用して大局的な処理構造を把握する．不具合調査に対しては③を利用して，ある制御内容の入力を起点とした前向き変数依存関係や，出力を起点とした後ろ向き変数依存関係を抽出し，その制御内容に関与する変数を特定して調査すれば良い[1][2].

ただし，従来のエンジニアリングツールには，解析結果同士を相互に作用させる機能までは備わっていない．例えば，異なるサブルーチンから共に参照・制御される変数の一覧を抽出するには，それぞれのサブルーチンに対して②を利用して得られる変数一覧について，それらの共通部分を人手で抽出する必要がある．

ある変数と別の変数の両方に影響される変数一覧についても，それぞれの変数を起点として③を利用して得られる２つの変数依存関係グラフについて，それらの両方に存在する変数を人手で抽出して一覧にする必要がある．

これら複合的な解析機能は，従来ツールへの追加実装で実現できる．しかしながら，機能を一品一様で実装する必要があり，ツール保守の観点で好ましくない．すなわち，従来のエンジニアリングツールの解析機能に係る解決すべき課題は，次のようにまとめることができる；

(1) 解析結果同士の論理演算の実現
(2) ある解析結果に対する別の解析機能の適用
(3) さまざまな解析機能を(1)(2)の組み合わせで実現しツールにおける一品一様な開発を回避

4. 条件組み合わせ可能なデータベース検索

解析結果同士を相互に作用させて得られる複合的な解析機能を，条件の組み合わせによって実現することが可能なラダー解析機能体系を確立するための参考技術として，データベース検索がある．

特許や図書などの文献は【書名】【著者】【分野】等の項目によりデータベース化することで，それら項目を指定した条件検索が可能となる．さらに異なる条件で検索した結果の論理演算 AND／OR／NOT により，条件を組み合わせた所望の検索結果を得ることができる．これを図式化すると図 5 のようになる．このようなデータベース検索では，検索対象も検索結果も同じデータ構造の集合になっているという特徴がある．

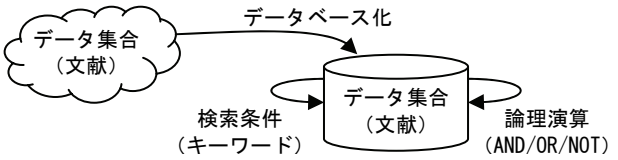

図 5：データベース化した文献データの検索

5. ラダー解析機能の体系化

本稿では，図 4 に示した従来のラダー解析機能を，図 5 に示した条件組み合わせ可能な機能体系にすることによって，課題(1)(2)(3)の解決を試みる．

すなわち，図 6 および表 1 に示すように，解析結果の出力データ構造と 1 対になるようにラダー解析機能を細分化して再定義する．そして，同じ出力データ構造どうしの論理演算機能を定義する．さらに，ある解析結果に対して別の解析機能を適用できるように，再定義した解析機能間の関連を整理する．このようにしてラダー解析機能を正規化することで，高度な解析を基礎的な機能の組み合わせで実現できる機能体系を確立する．

5.1 従来のラダー解析機能の細分化

従来の解析機能①②③は，解析対象プログラムをフォルダやファイルで指定するが，プログラムは複数の POU から構成されるため，ツール内部ではいずれも POU を単位とした解析処理が行われていた．

そこで新たに POU の集合を抽出する機能 PRG を定義し，POU における変数一覧抽出機能 DREF を定義することによって，②を PRG 機能と DREF 機能に細分化する．

同様に①を PRG 機能と FPOU／BPOU 機能に細分化する．ここで FPOU はある POU が呼び出す先の POU を再帰的に抽出する機能，BPOU はある POU を呼び出している元の POU を再帰的に抽出する機能である．

また③についても，PRG 機能と FSLC／BSLC 機能に細分化する．ここで FSLC はある POU 集合における前向きの変数依存関係抽出機能（該変数の値の変化によって影響されるすべての変数の抽出），BSLC はある POU 集合における後向きの変数依存関係抽出機能（該変数の値に影響を与えるすべての変数の抽出）である．

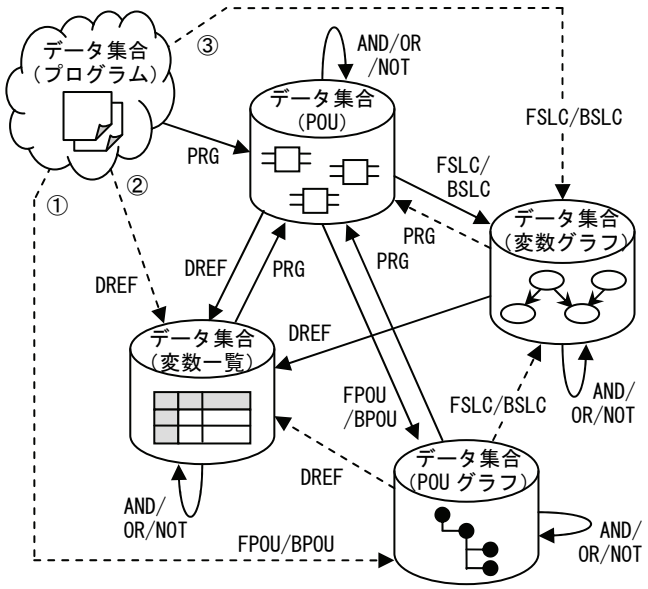

図 6：条件組み合わせ可能なラダー解析機能体系

表 1：ラダー解析機能の細分化定義

解析機能	解析対象	解析結果の出力データ構造
PRG POU の集合を抽出する	・プログラム ・変数一覧 ・変数依存関係有向グラフ ・POU 呼合関係有向グラフ	POU 集合
DREF 使用されている変数の一覧を抽出する	・プログラム ・POU 集合 ・変数依存関係有向グラフ ・POU 呼合関係有向グラフ	変数一覧
FSLC／BSLC 指定された変数を起点とした変数依存関係を抽出する	・プログラム ・POU 集合 ・POU 呼合関係有向グラフ	変数依存関係有向グラフ
FPOU／BPOU 指定された POU を起点とした呼び合い関係を抽出する	・プログラム ・POU 集合	POU 呼合関係有向グラフ
AND／OR／NOT 同じ形式の解析結果同士の論理演算を行う	・POU 集合 ・変数一覧 ・変数依存関係有向グラフ ・POU 呼合関係有向グラフ	解析対象と同じ

5.2 ラダー解析結果同士の論理演算

細分化して再定義した解析機能が出力する解析結果データ集合に対して，論理演算 AND／OR／NOT を機能定義する．これらの処理は，それぞれのデータ構造に応じた個別の実装が必要となるが，これによってどの解析結果に対しても同じ論理演算 AND／OR／NOT 機能を適用することが可能となる．

5.3 各ラダー解析機能間の関連整理

図 6 では，定義した解析機能のうち，個別の実装を必要とする解析機能を実線矢印で示し，実線矢印の処理の組み合わせで実現できる解析処理を破線矢印で示している．これについての詳細を以下に説明する．

PRG 機能　POU の集合は，プログラムを解析対象として抽出する以外に，変数一覧や変数依存関係グラフ，POU 呼び合い関係グラフを解析対象としても抽出できることから，これらをすべて同じ PRG と機能定義する．このうち，変数一覧を解析対象とした POU 集合の抽出は，各変数が使用されている POU を集計する処理となり，個別の実装を必要とする．また，POU 呼び合い関係グラフを解析対象とした POU 集合の抽出は，グラフに登場する POU を集計する処理となり，こちらも個別の実装を必要とする．しかしながら，変数依存関係グラフを解析対象とした POU 集合の抽出は，グラフに登場する変数を集計し，それら変数が使用されている POU を集計する処理となることから，先ず変数依存関係グラフを解析対象として（後述する）DREF を適用して変数一覧を抽出し，次にこの変数一覧を解析対象として PRG を適用することで処理が実現できるので，個別の実装を必要としない．すなわち，変数依存関係グラフを解析対象とした PRG 機能は，変数依存関係グラフを解析対象とした DREF と，その出力結果を解析対象とした PRG 機能の組み合わせで実現できる．

DREF 機能　変数一覧も同様に，プログラムや POU 集合を解析対象として抽出する以外に，変数依存関係グラフや POU 呼び合い関係グラフを解析対象としても抽出できることから，これらをすべて同じ DREF と機能定義する．プログラムや POU 集合を解析対象とした DREF は，5.1 で述べたとおりである．変数依存関係グラフを解析対象とした DREF は，グラフに登場する変数を集計する処理となり，個別の実装を必要とする．POU 呼び合い関係グラフを解析対象とした DREF は，グラフに登場する POU を集計し，それら POU で使用される変数を抽出する処理となることから，POU 呼び合い関係グラフを解析対象とした PRG 機能と，その出力結果を解析対象とした DREF 機能の組み合わせで実現できるので，個別の実装を必要としない．

FSLC／BSLC 機能　変数依存関係グラフについては，プログラムや POU 集合を解析対象として抽出する以外に，POU 呼び合い関係グラフを解析対象としても抽出できることから，これらをすべて同じ FSLC／BSLC と機能定義する．POU 集合を解析対象とした FSLC／BSLC は，5.1 で述べたとおりである．POU 呼び合い関係グラフを解析対象とした FSLC／BSLC は，先ず POU 呼び合い関係グラフを解析対象とした PRG 機能によって POU 集合を抽出し，次にこの POU 集合を解析対象として FSLC／BSLC 機能を適用することによって処理が実現できるので，個別の実装を必要としない．

ここで注意すべき点は，変数一覧を解析対象とした FSLC／BSLC は機能定義しないところである．その理由は，次のとおりである；

変数どうしの依存関係はプログラム記述によって決定づけられるものであるから，FSLC／BSLC 機能の解析対象はプログラム記述を有しているデータに限定される．従って，プログラムや POU 集合はもちろんのこと，POU 呼び合い関係の中で記述され決定づけられている変数どうしの依存関係を抽出する解析処理は，あってしかるべき機能といえる．しかしながら変数一覧は，変数依存関係グラフよりも情報量を絞った解析結果データであるから，変数依存関係グラフから変数一覧を抽出することはあっても，その逆の機能を定義することは不自然である．以上の理由により，

本稿では変数一覧を解析対象とした FSLC／BSLC を機能定義しないことにする．

FPOU／BPOU 機能　　POU 呼び合い関係グラフについては，プログラムや POU 集合を解析対象として抽出する FPOU／BPOU と機能定義する．これ以外の変数一覧や変数依存関係グラフを解析対象とした FPOU／BPOU は機能定義しない．その理由は，次のとおりである；

POU 呼び合い関係もまたプログラム記述によって決定づけられるものであるから，FPOU／BPOU 機能の解析対象はプログラム記述を有しているデータに限定される．従って，プログラムや POU 集合の中で記述され決定づけられている POU どうしの呼び合い関係を抽出する解析処理は，あってしかるべき機能といえる．しかしながら変数依存関係グラフや変数一覧は，POU 呼び合い関係グラフよりも情報量を絞った解析結果データであるから，POU 呼び合い関係グラフから変数依存関係グラフや変数一覧を抽出することはあっても，その逆の機能を定義することは不自然である．以上の理由により，本稿では変数一覧や変数依存関係グラフを解析対象とした FPOU／BPOU を機能定義しない．

6. 検索的解析支援ツールの開発

6.1 条件式入力ユーザインタフェース

本ツールは，解析条件を入力するユーザインタフェース部と，図 6 に基づいた解析処理を行う解析処理部から構成される（図 10 参照）．ユーザインタフェース部では，解析条件を条件式の形式で入力するとともに，その解析結果をデータ形式に応じて画面表示する．条件式は，解析コマンドおよび解析対象や解析条件を指定する引数から構成される．条件式は解析処理部にて処理される．

本ツールのユーザインタフェースを図 7 に示す．また，表 1 に基づいて定義した本ツールの解析コマンドを表 2 に示す．以下，図 7 に示した設定例を，具体的に説明する．

式番号 S001 は，PRG コマンドによって MAN というプログラムファイルから POU 集合を抽出し，その結果として POU が 1 つ得られている．

式番号 S002 は，DREF コマンドの引数に式番号 S001 と変数型 X を指定しており，式番号 S001 で得られた POU 集合を解析対象として，そこで使用されている X デバイス（入力変数）の一覧を抽出し，その結果として 248 個の X デバイスが得られている．式番号 S003 も同様に DREF を処理し，116 個の Y デバイスが得られている．

式番号 S004 は，式番号 S002 で得られた変数一覧と，式番号 S003 で得られた変数一覧の OR（和集合）を処理し，その結果として 364 個の変数一覧を抽出している．このように論理演算 AND／OR／NOT の 2 つの引数には，同じデータ構造を出力する式番号を指定する．表 2 に示す各コマンドの引数として指定された式番号が，表 1 で定義されていないデータ構造の場合には，エラーとなる．

式番号 S005 は，DREF コマンドの引数に式番号 S001 と変数型＊を指定しており，使用されている全デバイス（全ての変数）632 個を抽出している．

式番号 S006 は，FSLC コマンドの引数に式番号 S001 と変数 XC60 を指定しており，XC60 を起点とした前向きの変数依存関係グラフを抽出し，その結果として起点からの深さが 3 の有向グラフが得られている．式番号 S007 も同様に FSLC を処理し，深さ 3 の有向グラフが得られている．

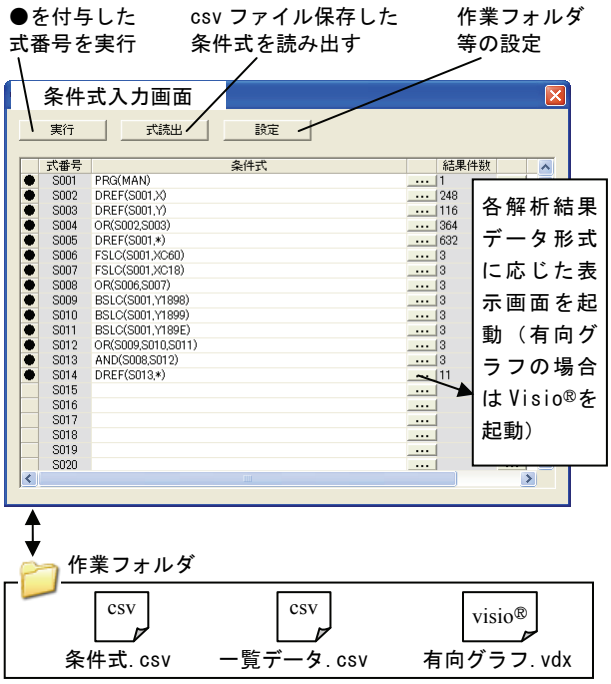

図 7：ユーザインタフェース（条件式入力画面）

表 2：解析コマンド定義

コマンド	第一引数	第二引数	出力データ形式
PRG	ファイル／式番号	N/A	POU 集合
DREF	ファイル／式番号	変数型	変数一覧
FSLC／BSLC	ファイル／式番号	起点変数	有向グラフ
FPOU／BPOU	ファイル／式番号	起点 POU	有向グラフ
AND／OR／NOT	式番号	式番号	引数に依存

式番号 S008 は，式番号 S006 および式番号 S007 でそれぞれ得られた変数依存関係グラフの OR（和集合）を処理し，その結果として深さ 3 の有向グラフを抽出している．

このように本検索的解析環境では，最初は解析対象となるプログラムをファイルで指定するが，その後は解析対象を式番号で指定し，ある解析結果データに対して別の解析コマンドを適用することで，任意に絞り込んだ解析結果が得られるようになっている．なお，本ツールは，解析条件式を CSV ファイルに保存し読み出せるようにしており，条件式入力画面によらず直接 CSV ファイルを編集しても良いようになっている．

6.2 有向グラフの可視化

本ツールでは図 8 に示すように Microsoft®の作図ソフトウェア Visio®（Visio はマイクロソフト社の商標）を用いて変数依存関係および POU 呼び合い関係の有向グラフを可視化している．得られた有向グラフのノードを Visio®シェイプとして出力すると共にシェイプ間を接続する矢印線を出力して，ノード間の依存関係や呼び合い関係の可視化を実現している．Visio®を用いた理由は次のとおりである；

有向グラフの可視化では，ノードの最適配置が問題となる．依存関係の起点からの深さに応じてノードを多段に配

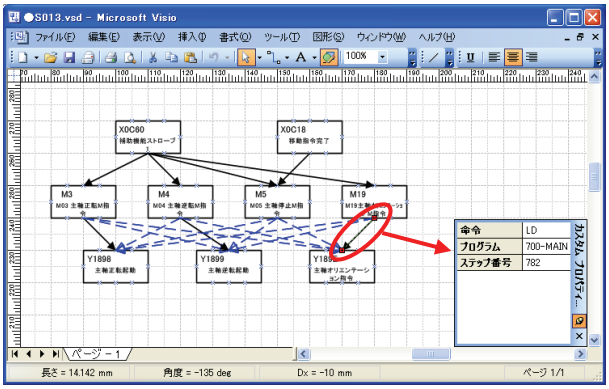

図 8：Visio®による有向グラフの可視化

置しつつ，関連の深いノードをグループ化する形で近くに配置すれば理解しやすい．ただし，どのようにノードをグループ化するかはアプリケーション依存度が大きく，ユーザにとって最も見やすいノード配置を自動的に決定する（つまり，ユーザにとって最も見やすいノード配置とは何かといった評価基準を整理する）ことは難しい課題である．有向グラフを Visio®で出力しておけばユーザは，ノード配置の気に入らないところについては，Visio®上でシェイプを移動させてアプリケーションに応じた理解しやすいノード配置へと自由に変更できる．

また Visio®では，シェイプに対して独自のデータテーブルを容易に定義追加できるという特徴がある．そして，シェイプ間を接続する矢印線もまたシェイプである．これら特徴を利用して，ノード間の依存関係を示す矢印線シェイプに，その依存関係が記述されている場所を合わせて出力することができる．これにより，有向グラフを構成する各ノードや依存関係矢印に該当するプログラム場所がどこであるかを直ちに確認することが可能となる．

図 8 は，変数依存関係グラフを可視化したものである．有向グラフのノードは変数であり，ノード間の矢印は変数間の依存関係を示す．当社のシーケンサでは，変数（デバイス）の内容を説明するための文字列（これをデバイスコメントと呼ぶ）を設定できるので，ここではノードとなっているシェイプのデータテーブルにデバイス名とデバイスコメントを持たせて，シェイプに表示している．また，依存関係を示す矢印線シェイプには，その依存関係が記述されているラダープログラム名とラダー行番号（ステップ番号），その行でその変数を制御しているラダー命令を持たせるようにしている．図 8 では，依存関係を示す矢印をクリックすると，右下のカスタムプロパティウィンドウに，その依存関係が 782 行目に記述されているといった情報が表示されている状態を示している．

6.3　ラダー解析例

大規模・複雑化したラダーにおいて，ある変数の依存関係を抽出するだけではグラフは発散的となる．これを解決するには，入力変数を起点とした FSLC と出力変数を起点とした BSLC を適宜組み合わせれば良い[8][9]．このようなラダー解析を，本ツールを用いて行った例を図 9 に示す．

ここでは，解析したい処理内容にとって入力となる変数が２つ，出力となる変数が３つあった場合に，解析したい処理内容に関連のある変数依存関係だけを絞り込んで抽出する例を示している．このような場合には，先ず２つの入

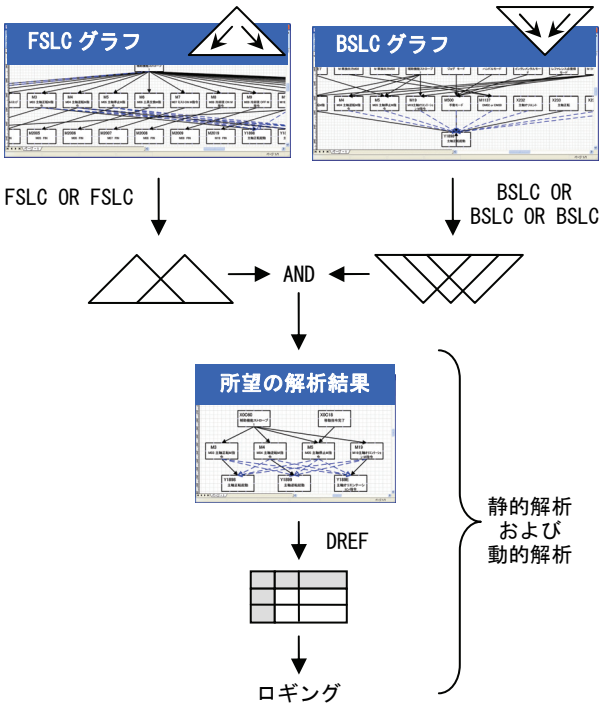

図 9：変数依存関係解析結果例

力変数を起点とした前向き依存関係有向グラフを FSLC でそれぞれ抽出し，その結果を OR した有向グラフを得る．次に，３つの出力変数を起点とした後ろ向き依存関係有向グラフを BSLC でそれぞれ抽出し，その結果を OR した有向グラフを得る．そして最後に，それら有向グラフの共通部分を AND 処理することによって，所望の解析結果を得る．さらに本ツールでは，抽出した変数依存関係グラフに存在する変数一覧の抽出も可能である．

図 9 に示した解析例は，あるシーケンス制御処理に不具合動作があったときに，依存関係の抽出といった静的解析に加え，解析によって抽出された変数の値が実行中にどのように変化するかを把握する動的解析も必要とされる場合に有効である．その処理に関連する入力変数および出力変数は，当該シーケンス制御の要求仕様に相当するものであるから，要求仕様に基づく解析条件によってラダーの静的解析を行い，その結果として得られた有向グラフから変数一覧を抽出し，それぞれの変数についてシーケンサ実行中の値の変化をロギングすることで，不具合動作に係る動的解析が可能となる．

このように本ツールを用いることで，大規模・複雑化したラダーであっても，不具合の原因となっている変数やプログラム場所を特定する解析作業が，従来に比べて格段に効率的に行えるようになった．

7. 考察

7.1　解析機能の API 化

四野見らは，プログラム解析ツールの解析結果にアクセスできる API を実現し，この API を使用することでさらに別のプログラム解析ツールの開発を容易にする試みを報告している[5]．例えばプログラムの ID を引数にとり，そのプログラムの中で使用される変数の一覧を返す API が用意されている．この API は，本研究におけるコマンド DREF

に相当する．四野見らの研究のねらいは本研究と同じである．本研究はプログラムの解析機能の正規化，すなわち API 化に関する研究でもある．

ただし，プログラム以外に解析結果も引数として取れるようにプログラム解析機能を体系的に整理してコマンド定義したところに本研究の意義がある．図 10 に示すように，本研究で開発したツールの実行処理部は，ユーザインタフェース部からコマンドを用いた条件式を受け取って解析処理を行う．この条件式コマンドを解析機能 API として利用すれば，目的・用途に特化したラダー解析機能の実現は容易なものとなり，ツールにおける一品一様な開発を回避できる．本ツールの条件式入力ユーザインタフェースの代わりに専用の条件入力画面を開発すれば，あとはその入力をコマンドに変換して実行処理部に渡すだけで良い．

7.2 解析機能の相互作用

大崎らは，プログラム理解を支援する依存解析ツールを試作している[3]．このツールには，得られた CFG(Control Flow Graph)や PDG(Program Dependence Graph)から節点を抽出する機能や，それら節点集合どうしの AND／OR／NOT 演算機能が備わる．これは本研究において，コマンド FSLC／BSLC で抽出した変数依存関係グラフに対してコマンド DREF を適用することに相当する．大崎らの研究では，CFG／PDG 抽出機能の下位機能として節点抽出機能が定義され，さらにその下位機能として AND／OR／NOT 集合演算機能が定義されており，ツールの操作形態（ユーザインタフェース）もこの機能定義に基づくものとなっている．

これに対して本研究では，条件式の組み合わせで絞り込みが可能なデータベース検索のようにプログラム解析できるユーザインタフェースにしている．すなわち，プログラム解析機能と集合演算機能を総合的に体系化してコマンド定義することによって，このようなユーザインタフェースが実現可能となっている点に意義がある．

7.3 より高度な解析支援に向けて

安原らは，C 言語のコールグラフを三次元に視覚化する VCRGL を提案している[4]．三次元による視覚化は操作方法に工夫が必要となるが，有向グラフの可視化手法として本研究で開発したツールでも今後検討すべき課題である．

石尾らは，特定の関心事に強く関連した要素だけを抽出できるようプログラムスライシング手法を拡張している[7]．巨大なシステムに対してプログラムスライシングを用いると，非常に多くの結果が出力され，開発者が理解しようとしている関心事と関連の弱いものが多く含まれるという課題は，石尾らの研究も本研究も同じである．石尾らの研究では，十分な知識がない場合でも使えるようにキーワードマッチと距離に基づいて結果を絞り込んでいる．このようなあいまい検索は，本研究で開発したツールにも今後取り入れてゆきたい機能である．

8．おわりに

本稿では，データベース検索のような条件の組み合わせを可能にするプログラム解析機能体系を確立した．また，これに基づいて検索的解析支援ツールを開発し，ラダーダイアグラムの解析に適用して，その有効性を確認した．

今後は 7.3 で考察した追加拡充機能を検討するとともに，他の分野で開発されるプログラムへの適用も検討したい．

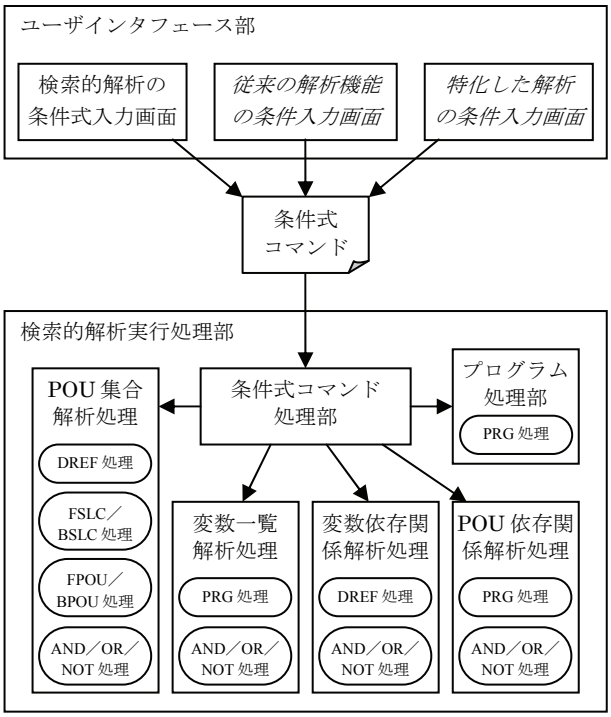

図 10：検索的解析支援ツールのアーキテクチャおよび実行処理部を利用した専用解析機能の実現

参考文献

[1] Jackson,D.and Rollings,E.J.: A new model of program dependences for reverse engineering, *SIGSOFT '94: Proceedings of the 2nd ACM SIGSOFT symposiumon Foundations of software engineering*, pp.2-10(1994)

[2] 下村：プログラムスライシング技術と応用, 共立出版 (1995)

[3] 大崎，山本，阿草：プログラム理解のための依存関係表示ツール，日本ソフトウェア科学会 FOSE'96, pp34-41 (1996)

[4] 安原, 山本, 阿草：オブジェクト属性を利用したソフトウェアの視覚化，日本ソフトウェア科学会 FOSE2000, pp198-196 (2000)

[5] 四野見，玉井：プログラム解析を提供する API の実現とその適用，コンピュータソフトウェア, Vol.22, No.1, pp91-97 (2005)

[6] 中村，藤本：シーケンス制御プログラムの構造解析に基づく機能モジュール抽出支援システムの研究，電気学会論文誌 D, Vol.127-D, No.10, pp1057-1063 (2007)

[7] 石尾，仁井谷，井上：プログラムスライシングを用いた機能的関心事の抽出，コンピュータソフトウェア, Vol.26, No.2, pp127-146 (2009)

[8] 仲井，金子：ラダーダイアグラムの静的構造解析に関する一考察，電子情報通信学会 2009 総合大会

[9] 仲井，佐藤：シーケンス制御プログラム解析支援環境の検討，SCI'10 第 54 回システム制御情報学会研究発表講演会 (2010)