(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号

特許第4757811号
(P4757811)

(45)発行日　平成23年8月24日(2011.8.24)　(24)登録日　平成23年6月10日(2011.6.10)

(51)Int.Cl.　　F I
G06F 9/48 (2006.01)　　G06F 9/06 610W
G06F 9/44 (2006.01)　　G06F 9/06 620A

請求項の数 9　(全 17 頁)

(21)出願番号　特願2007-38025(P2007-38025)
(22)出願日　平成19年2月19日(2007.2.19)
(65)公開番号　特開2008-204069(P2008-204069A)
(43)公開日　平成20年9月4日(2008.9.4)
審査請求日　平成20年3月10日(2008.3.10)
審判番号　不服2009-14928(P2009-14928/J1)
審判請求日　平成21年8月18日(2009.8.18)

(73)特許権者　000004237
日本電気株式会社
東京都港区芝五丁目７番１号
(74)代理人　100130029
弁理士　永井　道雄
(74)代理人　100166338
弁理士　関口　正夫
(74)代理人　100152054
弁理士　仲野　孝雅
(72)発明者　高井　真志
東京都港区芝五丁目７番１号　日本電気株式会社内

最終頁に続く

(54)【発明の名称】ジョブ制御言語により記述されたジョブ制御文からジョブネットワークフローを生成する装置及びその方法

(57)【特許請求の範囲】
【請求項１】
ＪＣＬファイルをジョブネットワークへ変換するための装置であるＪＣＬファイル／ジョブネットワーク変換装置が行う、ＪＣＬファイル／ジョブネットワーク変換方法において、
第１の記憶手段から、前記変換対象のＪＣＬファイル群を読み出すＪＣＬファイル入力ステップと、
各ＪＣＬファイルに含まれる各行を読み込み、当該各行に依存関係を指定する行が存在するか否かを確認し、当該確認の結果、当該ＪＣＬファイルの実行のために、事前に実行すべき他の１以上のＪＣＬファイルがあるとき、このような関係を前記他の１以上のＪＣＬファイルの論理積を前提部として持ち、当該ＪＣＬファイルを結論部として持った論理式で表して、その論理式を論理式集合に追加し、前記確認の結果、当該ＪＣＬファイルの実行のために、事前に実行すべき他のＪＣＬファイルがないとき、当該ＪＣＬファイルを、基底として初期の基底集合に追加する依存関係解析ステップと、
前記論理式集合に含まれる各論理式について、前記前提部に含まれる基底であるＪＣＬファイルが前記基底集合に含まれている場合には、当該基底であるＪＣＬファイルを前記前提部から削除し、前記前提部に含まれる全ての基底であるＪＣＬファイルが削除された論理式の結論部にあるＪＣＬファイルを新たな基底として前記基底集合に追加すると共に、当該論理式を前記論理式集合から削除する、といった処理を、ＪＣＬファイルの新たな基底としての前記基底集合への追加及び前記論理式の前提部からの基底であるＪＣＬファ

イルの削除が生じなくなるまで繰り返す依存関係解決ステップと、
前記依存関係を解決した基底を変換対象としてＪＣＬファイルをジョブネットワークに変換し、該ジョブネットワークを第２の記憶手段に格納するジョブネットワーク生成ステップと、
を備えることを特徴とするＪＣＬファイル／ジョブネットワーク変換方法。
【請求項２】
請求項１に記載のＪＣＬファイル／ジョブネットワーク変換方法において、
前記ジョブネットワーク生成ステップは、
前記依存関係が解決されている基底であるＪＣＬファイルからＪＣＬ命令を１つずつ読み出し、当該ＪＣＬ命令を、変換辞書において対応しているＯＳの命令に変換し、当該ＯＳ命令を含む中間ファイルを生成するＪＣＬ／中間ファイル変換ステップを含むことを特徴とするＪＣＬファイル／ジョブネットワーク変換方法。
【請求項３】
請求項２に記載のＪＣＬファイル／ジョブネットワーク変換方法において、
前記ジョブネットワーク生成ステップは、
前記中間ファイルを一行ずつ読み込み、コメントアウトされたＪＣＬ命令のうち順序制御にかかわるものを検出し、当該検出したＪＣＬ命令に応じた処理を行うことにより、中間ファイルを分割し、部品化し、ＪＣＬ命令をジョブネットワークの部品に変換し、ＯＳの命令は、ＯＳ実行命令とすることにより、中間ファイルをジョブネットワークフローに変換する中間ファイル／ジョブネットワーク変換ステップを更に含むことを特徴とするＪＣＬファイル／ジョブネットワーク変換方法。
【請求項４】
ＪＣＬファイルをジョブネットワークへ変換するための装置であるＪＣＬファイル／ジョブネットワーク変換装置において、
第１の記憶手段から、前記変換対象のＪＣＬファイル群を読み出すＪＣＬファイル入力手段と、
各ＪＣＬファイルに含まれる各行を読み込み、当該各行に依存関係を指定する行が存在するか否かを確認し、当該確認の結果、当該ＪＣＬファイルの実行のために、事前に実行すべき他の１以上のＪＣＬファイルがあるとき、このような関係を前記他の１以上のＪＣＬファイルの論理積を前提部として持ち、当該ＪＣＬファイルを結論部として持った論理式で表して、その論理式を論理式集合に追加し、前記確認の結果、当該ＪＣＬファイルの実行のために、事前に実行すべき他のＪＣＬファイルがないとき、当該ＪＣＬファイルを、基底として初期の基底集合に追加する依存関係解析手段と、
前記論理式集合に含まれる各論理式について、前記前提部に含まれる基底であるＪＣＬファイルが前記基底集合に含まれている場合には、当該基底であるＪＣＬファイルを前記前提部から削除し、前記前提部に含まれる全ての基底であるＪＣＬファイルが削除された論理式の結論部にあるＪＣＬファイルを新たな基底として前記基底集合に追加すると共に、当該論理式を前記論理式集合から削除する、といった処理を、ＪＣＬファイルの新たな基底としての前記基底集合への追加及び前記論理式の前提部からの基底であるＪＣＬファイルの削除が生じなくなるまで繰り返す依存関係解決手段と、
前記依存関係を解決した基底を変換対象としてＪＣＬファイルをジョブネットワークに変換し、該ジョブネットワークを第２の記憶手段に格納するジョブネットワーク生成手段と、
を備えることを特徴とするＪＣＬファイル／ジョブネットワーク変換装置。
【請求項５】
請求項４に記載のＪＣＬファイル／ジョブネットワーク変換装置において、
前記ジョブネットワーク生成手段は、
前記依存関係が解決されている基底であるＪＣＬファイルからＪＣＬ命令を１つずつ読み出し、当該ＪＣＬ命令を、変換辞書において対応しているＯＳの命令に変換し、当該ＯＳ命令を含む中間ファイルを生成するＪＣＬ／中間ファイル変換手段を含むことを特徴と

するＪＣＬファイル／ジョブネットワーク変換装置。
【請求項６】
請求項５に記載のＪＣＬファイル／ジョブネットワーク変換装置において、
前記ジョブネットワーク生成手段は、
前記中間ファイルを一行ずつ読み込み、コメントアウトされたＪＣＬ命令のうち順序制御にかかわるものを検出し、当該検出したＪＣＬ命令に応じた処理を行うことにより、中間ファイルを分割し、部品化し、ＪＣＬ命令をジョブネットワークの部品に変換し、ＯＳの命令は、ＯＳ実行命令とすることにより、中間ファイルをジョブネットワークフローに変換する中間ファイル／ジョブネットワーク変換手段を更に含むことを特徴とするＪＣＬファイル／ジョブネットワーク変換装置。
【請求項７】
ＪＣＬファイルをジョブネットワークへ変換するためのＪＣＬファイル／ジョブネットワーク変換方法に含まれる各ステップをコンピュータに実行させるためのＪＣＬファイル／ジョブネットワーク変換プログラムにおいて、
第１の記憶手段から、前記変換対象のＪＣＬファイル群を読み出すＪＣＬファイル入力ステップと、
各ＪＣＬファイルに含まれる各行を読み込み、当該各行に依存関係を指定する行が存在するか否かを確認し、当該確認の結果、当該ＪＣＬファイルの実行のために、事前に実行すべき他の１以上のＪＣＬファイルがあるとき、このような関係を前記他の１以上のＪＣＬファイルの論理積を前提部として持ち、当該ＪＣＬファイルを結論部として持った論理式で表して、その論理式を論理式集合に追加し、当該ＪＣＬファイルの実行のために、事前に実行すべき他のＪＣＬファイルがないとき、当該ＪＣＬファイルを、基底として初期の基底集合に追加する依存関係解析ステップと、
前記論理式集合に含まれる各論理式について、前記前提部に含まれる基底であるＪＣＬファイルが前記基底集合に含まれている場合には、当該基底であるＪＣＬファイルを前記前提部から削除し、前記前提部に含まれる全ての基底であるＪＣＬファイルが削除された論理式の結論部にあるＪＣＬファイルを新たな基底として前記基底集合に追加すると共に、当該論理式を前記論理式集合から削除する、といった処理を、ＪＣＬファイルの新たな基底としての前記基底集合への追加及び前記論理式の前提部からの基底であるＪＣＬファイルの削除が生じなくなるまで繰り返す依存関係解決ステップと、
前記依存関係を解決した基底を変換対象としてＪＣＬファイルをジョブネットワークに変換し、該ジョブネットワークを第２の記憶手段に格納するジョブネットワーク生成ステップと、
をコンピュータに実行させるためのＪＣＬファイル／ジョブネットワーク変換プログラム。
【請求項８】
請求項７に記載のＪＣＬファイル／ジョブネットワーク変換プログラムにおいて、
前記ジョブネットワーク生成ステップは、
前記依存関係が解決されている基底であるＪＣＬファイルからＪＣＬ命令を１つずつ読み出し、当該ＪＣＬ命令を、変換辞書において対応しているＯＳの命令に変換し、当該ＯＳ命令を含む中間ファイルを生成するＪＣＬ／中間ファイル変換ステップを含むことを特徴とするＪＣＬファイル／ジョブネットワーク変換プログラム。
【請求項９】
請求項８に記載のＪＣＬファイル／ジョブネットワーク変換プログラムにおいて、
前記ジョブネットワーク生成ステップは、
前記中間ファイルを一行ずつ読み込み、コメントアウトされたＪＣＬ命令のうち順序制御にかかわるものを検出し、当該検出したＪＣＬ命令に応じた処理を行うことにより、中間ファイルを分割し、部品化し、ＪＣＬ命令をジョブネットワークの部品に変換し、ＯＳの命令は、ＯＳ実行命令とすることにより、中間ファイルをジョブネットワークフローに変換する中間ファイル／ジョブネットワーク変換ステップを更に含むことを特徴とするＪ

ＣＬファイル／ジョブネットワーク変換プログラム。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、ジョブ制御言語により記述されたジョブ制御文からジョブネットワークフローを生成する装置、その装置及びそのプログラムに関する。
【背景技術】
【０００２】
近年、多くの企業システムにおいて、メインフレームからＵＮＩＸ（登録商標）サーバやＷｉｎｄｏｗｓ（登録商標）などの汎用ＯＳを使用したオープンシステムへのシステム更改が活発に行われている。
【０００３】
しかし、そのようなシステム更改においては、これまでメインフレーム上で運用されていたジョブ資産の移行が大きな問題となる。
【０００４】
通常、それらのジョブ資産は、ジョブ制御言語（ＪＣＬ）で記述されているが、オープンシステム上ではこの言語を利用できないため、ＵＮＩＸ（登録商標）のシェルスクリプト等の言語にそれらを変換し、さらにＪＣＬの制御命令をジョブネットワークのフローに置き換える必要がある。
【０００５】
通常このような、ＪＣＬからジョブネットワークへの変換はＳＥの人手で行われてきた。
【０００６】
しかし、数千のＪＣＬを、ジョブネットワークに手動で作り変える作業は、多大な工数を必要とする。また、人手での作業は、手動ゆえのバグの作りこみが多発し、更にそれを検出するためのテストにより、移行コストは膨大なものとなる。
【特許文献１】特開平１１－０２４９１３号公報
【発明の開示】
【発明が解決しようとする課題】
【０００７】
第１の問題点は、手動でのＪＣＬからジョブネットワークへの移行は、人為的ミスによる品質低下と、それをリカバリするために実施されるテスト作業により、膨大なコストを消費することになることである。
【０００８】
第２の問題点は、そもそもＪＣＬファイルにはファイル間にさまざまな依存関係、たとえば、あるＪＣＬの実行のためには、あるＪＣＬの実行が終了している必要がある等があるが、たとえば、ＪＣＬファイルが数千個存在するような環境では、その依存関係の存在や、依存関係の内容を完全に把握している人材は少ない。そのため、この依存関係の調査などに多くのコストを消費することになる。これが更なるコストとリスクとなって、移行作業にかかることになる。
【０００９】
更に、依存関係のため、例え簡易変換ツールを作成しても、実際のテスト時には依存関係によるエラー多発することになり、結局、人手での原因究明および修正が必要となり、コスト削減効果が大きく減じられていた。
【００１０】
そこで、本発明は、人手によらずに、ＪＣＬファイルをジョブネットワークに変換することを目的とする。
【課題を解決するための手段】
【００１１】
本発明によれば、ＪＣＬファイルをジョブネットワークへ変換するための装置であるＪ

ＣＬファイル／ジョブネットワーク変換装置が行う、ＪＣＬファイル／ジョブネットワーク変換方法において、第１の記憶手段から、前記変換対象のＪＣＬファイル群を読み出すＪＣＬファイル入力ステップと、各ＪＣＬファイルに含まれる各行を読み込み、当該各行に依存関係を指定する行が存在するか否かを確認し、当該確認の結果、当該ＪＣＬファイルの実行のために、事前に実行すべき他の１以上のＪＣＬファイルがあるとき、このような関係を前記他の１以上のＪＣＬファイルの論理積を前提部として持ち、当該ＪＣＬファイルを結論部として持った論理式で表して、その論理式を論理式集合に追加し、前記確認の結果、当該ＪＣＬファイルの実行のために、事前に実行すべき他のＪＣＬファイルがないとき、当該ＪＣＬファイルを、基底として初期の基底集合に追加する依存関係解析ステップと、前記論理式集合に含まれる各論理式について、前記前提部に含まれる基底であるＪＣＬファイルが前記基底集合に含まれている場合には、当該基底であるＪＣＬファイルを前記前提部から削除し、前記前提部に含まれる全ての基底であるＪＣＬファイルが削除された論理式の結論部にあるＪＣＬファイルを新たな基底として前記基底集合に追加すると共に、当該論理式を前記論理式集合から削除する、といった処理を、ＪＣＬファイルの新たな基底としての前記基底集合への追加及び前記論理式の前提部からの基底であるＪＣＬファイルの削除が生じなくなるまで繰り返す依存関係解決ステップと、前記依存関係を解決した基底を変換対象としてＪＣＬファイルをジョブネットワークに変換し、該ジョブネットワークを第２の記憶手段に格納するジョブネットワーク生成ステップと、を備えることを特徴とするＪＣＬファイル／ジョブネットワーク変換方法が提供される。

【００１２】

また、本発明によれば、ＪＣＬファイルをジョブネットワークへ変換するための装置であるＪＣＬファイル／ジョブネットワーク変換装置において、第１の記憶手段から、前記変換対象のＪＣＬファイル群を読み出すＪＣＬファイル入力手段と、各ＪＣＬファイルに含まれる各行を読み込み、当該各行に依存関係を指定する行が存在するか否かを確認し、当該確認の結果、当該ＪＣＬファイルの実行のために、事前に実行すべき他の１以上のＪＣＬファイルがあるとき、このような関係を前記他の１以上のＪＣＬファイルの論理積を前提部として持ち、当該ＪＣＬファイルを結論部として持った論理式で表して、その論理式を論理式集合に追加し、前記確認の結果、当該ＪＣＬファイルの実行のために、事前に実行すべき他のＪＣＬファイルがないとき、当該ＪＣＬファイルを、基底として初期の基底集合に追加する依存関係解析手段と、前記論理式集合に含まれる各論理式について、前記前提部に含まれる基底であるＪＣＬファイルが前記基底集合に含まれている場合には、当該基底であるＪＣＬファイルを前記前提部から削除し、前記前提部に含まれる全ての基底であるＪＣＬファイルが削除された論理式の結論部にあるＪＣＬファイルを新たな基底として前記基底集合に追加すると共に、当該論理式を前記論理式集合から削除する、といった処理を、ＪＣＬファイルの新たな基底としての前記基底集合への追加及び前記論理式の前提部からの基底であるＪＣＬファイルの削除が生じなくなるまで繰り返す依存関係解決手段と、前記依存関係を解決した基底を変換対象としてＪＣＬファイルをジョブネットワークに変換し、該ジョブネットワークを第２の記憶手段に格納するジョブネットワーク生成手段と、を備えることを特徴とするＪＣＬファイル／ジョブネットワーク変換装置が提供される。

【００１３】

更に、本発明によれば、ＪＣＬファイルをジョブネットワークへ変換するためのＪＣＬファイル／ジョブネットワーク変換方法に含まれる各ステップをコンピュータに実行させるためのＪＣＬファイル／ジョブネットワーク変換プログラムにおいて、第１の記憶手段から、前記変換対象のＪＣＬファイル群を読み出すＪＣＬファイル入力ステップと、各ＪＣＬファイルに含まれる各行を読み込み、当該各行に依存関係を指定する行が存在するか否かを確認し、当該確認の結果、当該ＪＣＬファイルの実行のために、事前に実行すべき他の１以上のＪＣＬファイルがあるとき、このような関係を前記他の１以上のＪＣＬファイルの論理積を前提部として持ち、当該ＪＣＬファイルを結論部として持った論理式で表して、その論理式を論理式集合に追加し、当該ＪＣＬファイルの実行のために、事前に実

行すべき他のＪＣＬファイルがないとき、当該ＪＣＬファイルを、基底として初期の基底集合に追加する依存関係解析ステップと、前記論理式集合に含まれる各論理式について、前記前提部に含まれる基底であるＪＣＬファイルが前記基底集合に含まれている場合には、当該基底であるＪＣＬファイルを前記前提部から削除し、前記前提部に含まれる全ての基底であるＪＣＬファイルが削除された論理式の結論部にあるＪＣＬファイルを新たな基底として前記基底集合に追加すると共に、当該論理式を前記論理式集合から削除する、といった処理を、ＪＣＬファイルの新たな基底としての前記基底集合への追加及び前記論理式の前提部からの基底であるＪＣＬファイルの削除が生じなくなるまで繰り返す依存関係解決ステップと、前記依存関係を解決した基底を変換対象としてＪＣＬファイルをジョブネットワークに変換し、該ジョブネットワークを第２の記憶手段に格納するジョブネットワーク生成ステップと、をコンピュータに実行させるためのＪＣＬファイル／ジョブネットワーク変換プログラムが提供される。

【発明の効果】

【００１６】

本発明によれば、人手によらずに、ＪＣＬファイルをジョブネットワークに変換することができるので、変換に要するコストを削減することができる。

【発明を実施するための最良の形態】

【００１７】

以下、図面を参照して本発明を実施するための最良の形態について詳細に説明する。

【００１８】

図１の第１の記憶装置１０１は変換対象のＪＣＬファイル群１０２が格納されている記憶装置である。

【００１９】

ＪＣＬファイル入力部１０３は、第１の記憶装置１０１から対象となるＪＣＬファイル群を読み出し、依存関係解析器１０５に送る。

【００２０】

依存関係解析器１０５は、送られたＪＣＬファイルを１つずつ走査し、各ＪＣＬファイルの依存関係の存在とその内容を検査し、それらをメモリ上に展開する。その際、依存関係を持たないＪＣＬファイルは、１０７（図２）のように推論のための基底（事実）の集合として利用する。逆に、依存関係を持つＪＣＬファイルは、その依存関係を論理式として表現される。例えば、依存関係解析の結果として、「ＪＣＬファイルＲ１の実行のためには、事前にＪＣＬファイルＢ１とＪＣＬファイルＢ２とＪＣＬファイルＢ３の実行が必要がある。」という依存関係が検出された場合、この依存関係を、

Ｂ１∧Ｂ２∧Ｂ３→Ｒ１

と表現する。このような依存関係の集合を１０７（図１及び図２参照）のように論理式集合として表現する。依存関係解析器１０５は２つの集合（基底（事実）ファイル集合１０６及び論理式（未解決）ファイル集合１０７）を依存関係解決器１０８に送る。

【００２１】

依存関係解決器１０８の概念図を図２に示す。

【００２２】

依存関係解決器１０８は、基底ファイル集合１０６から１つの基底ファイルを取り出し、それを論理式ファイル集合１０７に適用し、推論を行う。

【００２３】

例えば、

Ｂ１∧Ｂ２∧Ｂ３→Ｒ１

という論理式があり、Ｂ１という事実（基底）が基底集合に存在した場合、上記論理式は（Ｂ１であることは自明であるとして）自動推論により、

Ｂ２∧Ｂ３→Ｒ１

という論理式に省略される。このような自動推論を論理式（未解決）ファイル集合１０７の全ての要素（論理式）に適用した後、完全に充足された結果（Ｒｅｓｕｌｔ）、例えば

、
Ｂ２→Ｒ１
という論理式に、Ｂ２が適用されれば、Ｒ１という結果（Ｒｅｓｕｌｔ）が得られるが、このような結果（Ｒｅｓｕｌｔ）を新たに判明した事実（基底）として、基底ファイル集合１０６に加え、以後の推論の基底として利用する。

【００２４】
このような自動推論を基底ファイル集合１０６の全ての要素に関して繰り返す。その結果、得られた基底ファイル集合１０６の各要素であるＪＣＬファイルは、依存関係を全く持たないもしくは依存関係が完全に解決されていることが、論理的に保証されている（推論の完全性の保証）。

【００２５】
ＪＣＬファイル依存関係出力部１０９は、自動推論の結果として得られた、基底ファイル集合１０６であるＪＣＬファイル群をＪＣＬ／中間ファイル変換器１１１に送る。更に、論理式（未解決）ファイル集合１０７の要素に関して、依存関係に問題がある旨の警告を表示させる。

【００２６】
本発明の全体構成が図１に示されている。

【００２７】
図１に示すように、本実施例は、変換対象のＪＣＬファイル群を格納する第１の記憶装置１０１と、第１の記憶装置１０１から変換対象となるＪＣＬファイル群を読み出すＪＣＬファイル入力部１０３と、ＪＣＬファイル群を解析し、依存関係を把握し、それを２種類の集合に集約し、それをメモリ上に展開する依存関係解析器１０５と、それらを使用して自動推論を行う依存関係解決器１０８と、自動推論の結果として得られた、基底集合をＪＣＬ／中間ファイル変換器１１１に送り、論理式集合の要素に関して、警告を表示するＪＣＬファイル依存関係出力部１０９と、ＪＣＬファイル依存関係出力部１０９から送られたＪＣＬファイル群を中間ファイル群に変換するＪＣＬ／中間ファイル変換器１１１と、中間ファイルをジョブネットワークに変換して、第２の記憶装置１１５に格納する中間ファイル／ジョブネットワーク変換器１１３によって構成される。

【００２８】
図３にＪＣＬ／中間ファイル変換器１１１の構成を示す。ＪＣＬ／中間ファイル変換器１１１は、ＪＣＬファイルを記憶装置から取り出すＪＣＬファイル入力部１１１－１と、受け取ったＪＣＬファイルを中間ファイルに変換する中間ファイル生成部１１１－３と、生成された中間ファイルを記憶装置に格納する中間ファイル出力部１１１－５を備える。また、中間ファイル生成部１１１－３はあらかじめ準備された変換辞書１１１－７を持つ。変換辞書１１１－７には、あらかじめＪＣＬとシェルスクリプトへの対応表が格納されており、中間ファイル生成部は、この辞書を参照しながら、変換処理を行っていく。

【００２９】
図４に中間ファイル／ジョブネットワーク変換器１１３の構成を示す。中間ファイル／ジョブネットワーク変換器１１３は、必要な中間ファイル群を記憶装置から取り出す中間ファイル入力部１１３－１と、中間ファイルを解析して、ジョブネットワークに変換する中間ファイル解析部１１３－３と、ファイルシステムに結果出力を行うジョブネットワーク出力部１１３－５で構成される。

【００３０】
図１の第１の記憶装置１０１は変換対象のＪＣＬファイル群が格納されている記憶装置である。

【００３１】
図１のＪＣＬファイル入力部１０３は、第１の記憶装置１０１から、対象となるＪＣＬファイル群を読み出し、依存関係解析器１０５に送る。

【００３２】
依存関係解析器１０５の動作を図５に示す。

【００３３】
依存関係解析器１０５は、送られたＪＣＬファイル集合を１つずつ走査し（ステップＳ２０１、Ｓ２０３、Ｓ２０５、各ファイルの依存関係の有無を調査する（ステップＳ２０７、Ｓ２０９、Ｓ２１１、Ｓ２１３、Ｓ２１５）。
【００３４】
当該ファイルに依存関係が存在しなければ、図６のような基底（事実）集合として登録される（ステップＳ２２１）。実際のメモリ上では図７のようなリスト構造で表現される。
【００３５】
当該ファイルに依存関係が存在した場合は、その依存関係は、図８のような論理式として解釈され、当該ファイルは論理式集合として、依存関係を含めて登録される（ステップＳ２１７）。
【００３６】
例えば、「ＪＣＬファイルＲ１の実行のためには、ＪＣＬファイルＤ１とＪＣＬファイルＤ２とＪＣＬファイルＤ３の実行が終了している必要がある。」という依存関係が検出された場合、この依存関係を、
Ｄ１ΛＤ２ΛＤ３→Ｒ１
と表現する。実際のメモリ上では、図９のようなリスト構造で表現される。
【００３７】
依存関係解析器１０５は、これら２種類の集合（基底集合と論理式集合）を、依存関係解決器１０８に送る。
【００３８】
依存関係解決器１０８の動作概念を図２に示す。依存関係解決器１０８は、基底ファイル集合１０６から１つの基底（ファイル）を取り出し、それを論理式ファイル集合１０７に適用し、推論を行う。図１、図２に示す依存関係解決器１０８の具体的な動作を図１０、図１１、図１２に示す。
【００３９】
依存関係解決器は、依存関係解析器１０５から入力された基底集合から１つの基底（ファイル）を取り出し、それを論理式集合に適用しようと試みる（ステップＳ２４１、Ｓ２４３、Ｓ２４５、Ｓ２４７、Ｓ２４９）。ステップ２４７の論理式リスト解決処理の詳細を図１１に示す。依存関係解決器１０８は、論理式リストから論理式を１つ取り出し、その論理式に対して、当該基底を使用して論理式の推論を試みる（ステップＳ２６３、Ｓ２６５、Ｓ２６７）。
【００４０】
論理式推論処理がステップ２６７であり、その動作の詳細を図１２に示す。図９のように各論理式は依存関係にあるファイルのリストを持っている。このリストから依存関係を１つ取り出し、その依存関係に当該基底が適用できるかどうかを試みる（ステップＳ２９１、Ｓ２９３、Ｓ２９５、Ｓ２９７）。
【００４１】
もし、基底が適用できれば、その依存関係は解決されたものとして、その依存関係をリストから削除する（ステップＳ２９９）。もしその結果、全ての依存関係が解決されたのであれば、その論理式は完全に充足されたものとして、その結果（ＪＣＬファイル）を論理式推論処理終了（充足）として、ステップ２６７を終了し、ステップ２６９に進む。ステップ２６９では、ステップ２６７の結果が充足であれば、その論理式をリストから削除し、当該論理式の結果を基底リストの末尾に加える（ステップＳ２７２、Ｓ２７３）。これで１つの論理式に関する推論が終わり、次の論理式の推論へ進む。以降、これを全ての論理式について繰り返す（ステップＳ２６３、Ｓ２６５、Ｓ２６７、Ｓ２６９、Ｓ２７１、Ｓ２７３、Ｓ２７５）。全ての論理式リストの要素についての推論が終了したら、論理式リスト解決処理終了として、図１０のステップ２４９に進み、改めて基底のピックアップを行う。

【００４２】
以上の推論の流れについて、図１３を例にとって説明する。図１３のような場合、ピックアップされた基底がＢ１であれば、ステップＡにより、依存関係Ｂ１が削除され、論理式を省略できる。さらに、次にピックアップされた基底がＢ２であれば、ステップＢが実行され、同様に依存関係Ｂ２が削除できる。最後にピックアップされた基底がＢ３であれば、ステップＣにより結果（Ｒｅｓｕｌｔ）Ｒ１が得られる。これが、新たな基底として、基底集合の末尾に加えられる。
【００４３】
このような自動推論を、図１０のように基底集合の全ての要素に関して繰り返す。その結果、得られた最終的な基底集合の各要素（ＪＣＬファイル）は、依存関係を全く持たないもしくは依存関係が完全に解決されていることが、論理的に保証されたファイル群であり、このファイル群に関しては、依存関係に関して、如何なるエラーも発生しないことが保証される。よって、ＪＣＬファイル依存関係出力部１０９は、このリストをＪＣＬ／中間ファイル変換器１１１に引き渡す。
【００４４】
逆に最終的な論理式集合の各要素（ＪＣＬファイル）は、依存関係により、問題が発生することが自明であるため、ＪＣＬファイル依存関係出力部１０９は、変換を行う前に、警告をオペレータに対して行う。
【００４５】
ＪＣＬ／中間ファイル変換器１１１の動作について、図３を参照して説明をする。ＪＣＬファイル入力部１１１－１は、ＪＣＬファイル依存関係出力部１０９から受け取った変換対象ＪＣＬファイルのリストを元に、第１の記憶装置１０１から変換対象のＪＣＬファイルを読み出す。中間ファイル生成部１１１－３は、変換辞書１１１－７を参照しながら、中間ファイルへの変換を行っていく。変換辞書には変換対象ＪＣＬ命令に対応する各ＯＳの命令が格納されている。中間ファイル生成部１１１－３は、図１４のフローチャートのように処理を行う。まず、ＪＣＬを一行ずつ読み込み、対象行と変換辞書を比較する（ステップＳ３１３、Ｓ３１５、Ｓ３１７、Ｓ３１９）。該当すれば、変換を行い、辞書の検索結果を中間ファイルに出力し、該当しなければ対象行をそのまま出力する（ステップＳ３２３、Ｓ３２５）。このとき、変換対象行のＪＣＬ命令は、その後のフロー変換のための入力行として、コメント行（例：ＵＮＩＸ（登録商標）では”＃”行）としてコメントアウトされる。この処理をＪＣＬのファイルのＥＯＦ（ＥｎｄＯｆＦｉｌｅ）まで繰り返す（ステップＳ３１３、Ｓ３１５、Ｓ３１７、Ｓ３１９、Ｓ３２１、Ｓ３２３、Ｓ３２５）。ＥＯＦに到達したら、作成された中間ファイルを変換結果として、中間ファイル／ジョブネットワーク変換器１１３７へと送る。実際にＪＣＬ／中間ファイル変換を行った例を図１５に示す。４０１のＪＣＬに対して図１４の処理を行うことにより、中間ファイル４０３が得られる。通常のＪＣＬ行（実処理１（ＪＣＬ）～実処理４（ＪＣＬ））は、変換先ＯＳに適合したシェルスクリプト（この例ではＵＮＩＸ（登録商標）シェル）の記述に変換される。また、順序関係を表す特定のＪＣＬ行（￥ＪＯＢ，￥ＳＵＢＪＯＢ等）はコメントとして残されている。
【００４６】
中間ファイル／ジョブネットワーク変換器１１３の動作を、図４を参照して説明する。中間ファイル入力部１１３－１はＪＣＬ／中間ファイル変換器１１１から受け取った中間ファイル群を、中間ファイル解析部１１３－３に送る。中間ファイル解析部１１３－３の動作を図１６、図１７、図１８、図１９、図２０のフローチャートに示す。
【００４７】
図１６が初動作である。中間ファイルを一行ずつ読み込み（Ｓ５０３）、コメントアウトされたＪＣＬ命令のうち、順序制御にかかわるものを検出する。まず、ＪＣＬの開始行である￥ＪＯＢ行を探す（Ｓ５０７）。￥ＪＯＢを検出すると、詳細解析処理を行う。図１７が詳細解析処理の動作を示す。￥ＳＵＢＪＯＢ、￥ＷＡＩＴＳＵＢ、￥ＷＡＩＴＳＵＢ、￥ＥＮＤＪＯＢ、￥ＲＵＮなどのＪＣＬの特定命令行をコメントアウトされたＪＣＬ

命令から検出し、それに応じた処理を行い、対応したジョブネットワーク部品をリンク構造としてメモリ上に登録していく。￥ＥＮＤＪＯＢを検出すると、メモリ上のリンクをクローズする。実際に中間ファイル／ジョブネットワーク変換を行った例を図１５に示す。中間ファイル４０３を上記のフローチャートに従い変換すると４０５のリンク構造が得られる。実処理１～３はＪＯＢ１～ＪＯＢ４のそれぞれの構造体の要素となっている。

【００４８】

ジョブネットワーク出力部１１３－５はメモリ上に展開されたリンクをフローとして出力する。図２０がその動作を示す。リンク構造で表現されたものはすでにフロー図と等価になっているので、そのまま描画／ファイル出力することができるかどうかを簡単にチェックするだけで、そのままジョブネットワークフローとして出力する。

【００４９】

本実施形態によれば、下記のような効果が得られる。

【００５０】

第一の効果は、本プログラムにより、ＪＣＬからジョブネットワーク等への移行作業において、ＪＣＬファイル間の依存関係によるリスクが極小となり、テスト工数が大幅に削減可能となることである。

【００５１】

その理由は、本プログラムにより自動生成されたジョブネットワークは、全く持たないもしくは依存関係が完全に解決されていることが、論理的に保証されているからである。

【００５２】

第二の効果は、本プログラムにより、ＪＣＬからジョブネットワークの完全な自動変換が可能になることによって、メインフレームからオープンシステムへのジョブ環境移行のコストおよびリスクが大幅に削減されることである。

【００５３】

その理由は、ＪＣＬ→ジョブネットワーク自動変換において最大障害であった、ファイル間の依存関係の問題が解消されるからである。

【図面の簡単な説明】

【００５４】

【図１】本発明の実施形態によるジョブ制御言語からジョブネットワークを生成する装置の構成を示すブロック図である。

【図２】図１に示す依存関係解決器の構成を示すブロック図である。

【図３】図１に示すＪＣＬ／中間ファイル変換器の構成を示すブロック図である。

【図４】図１に示す中間ファイル／ジョブネットワーク変換器の構成を示すブロック図である。

【図５】図１に示す依存関係解決器の動作を示すフローチャートである。

【図６】本発明の実施形態による基底（事実）集合を示す図である。

【図７】本発明の実施形態による基底集合リストを示す図である。

【図８】本発明の実施形態による論理式（未解決）集合を示す図である。

【図９】本発明の実施形態による論理式集合リストを示す図である。

【図１０】図１に示す依存関係解決器の動作を示すフローチャート（１／３）である。

【図１１】図１に示す依存関係解決器の動作を示すフローチャート（２／３）である。

【図１２】図１に示す依存関係解決器の動作を示すフローチャート（３／３）である。

【図１３】本発明の実施形態による基底集合による推論解決を示す図である。

【図１４】図３に示す中間ファイル生成部の動作を示すフローチャートである。

【図１５】ＪＣＬから中間ファイルを生成し、更に、ジョブネットワークを生成することを示す図である。

【図１６】図４に示す中間ファイル解析部の動作を示すフローチャート（１／５）である。

【図１７】図４に示す中間ファイル解析部の動作を示すフローチャート（２／５）である。

【図１８】図４に示す中間ファイル解析部の動作を示すフローチャート（３／５）である。

【図１９】図４に示す中間ファイル解析部の動作を示すフローチャート（４／５）である。

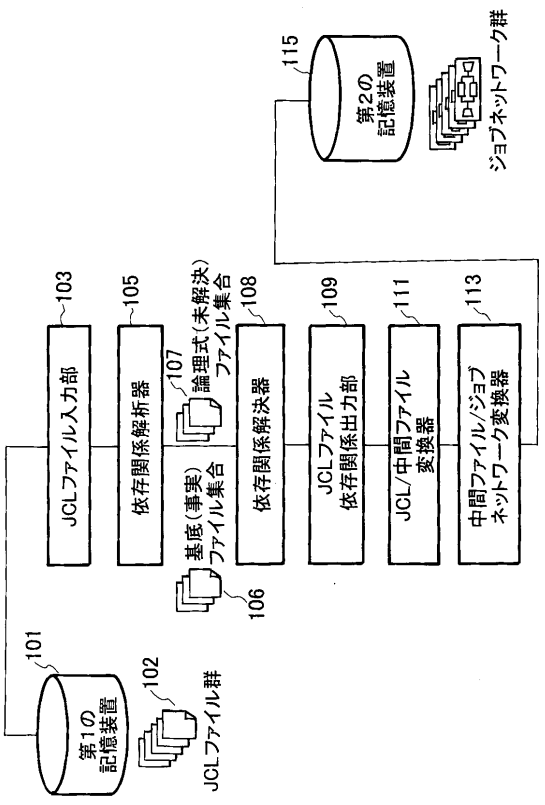
【図２０】図４に示す中間ファイル解析部の動作を示すフローチャート（５／５）である。

【符号の説明】

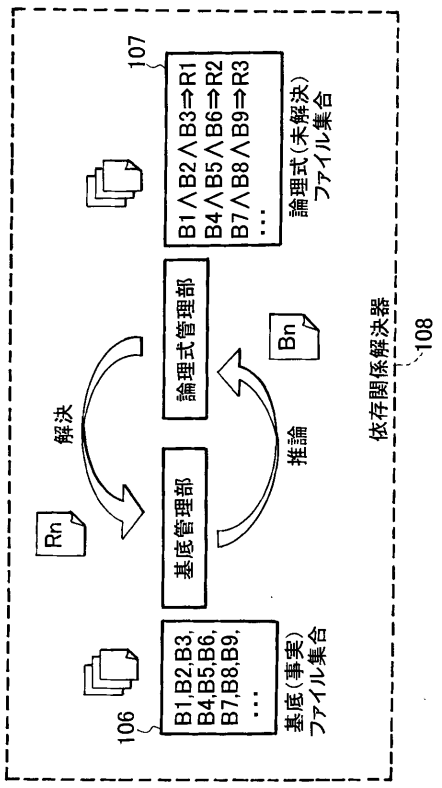
【００５５】

１０１　第１の記憶部
１０３　ＪＣＬファイル入力部
１０５　依存関係解析器
１０８　依存関係解決器
１０９　ＪＣＬファイル依存関係出力部
１１１　ＪＣＬ／中間ファイル変換器
１１３　中間ファイル／ジョブネットワーク変換器
１１５　第２の記憶装置

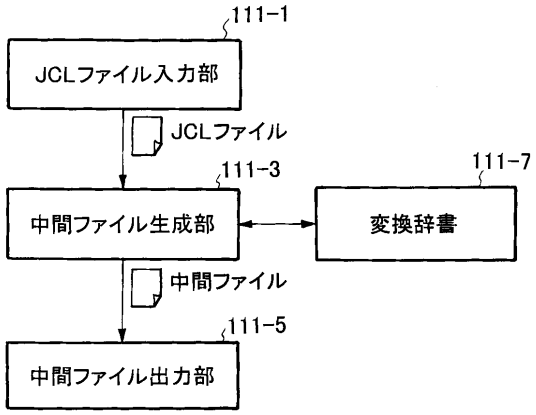
【図１】



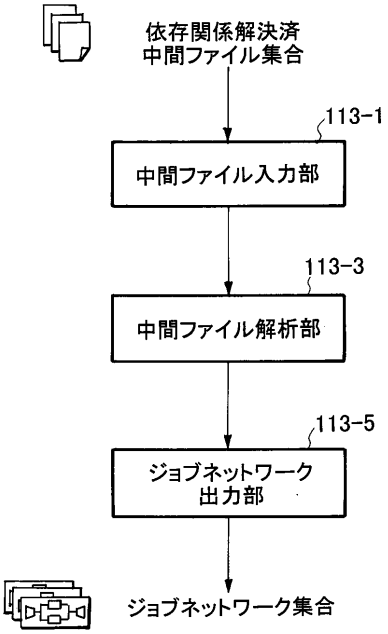
【図２】



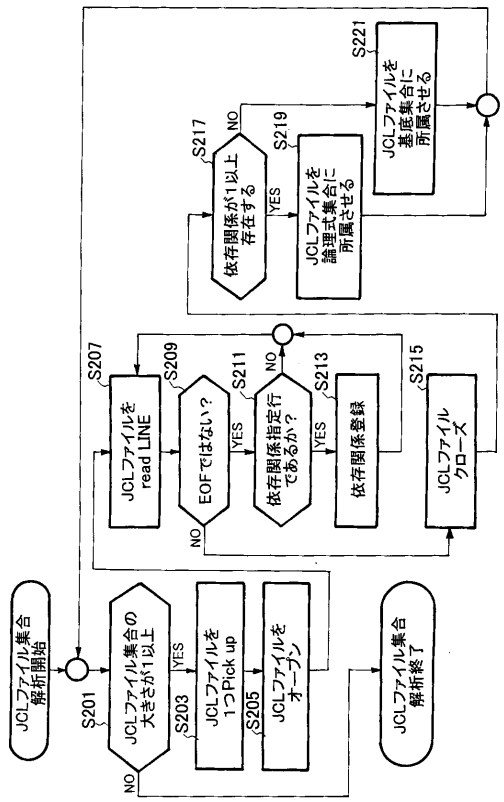
【図３】


111-1
JCLファイル入力部
JCLファイル
111-3
111-7
中間ファイル生成部
変換辞書
中間ファイル
111-5
中間ファイル出力部

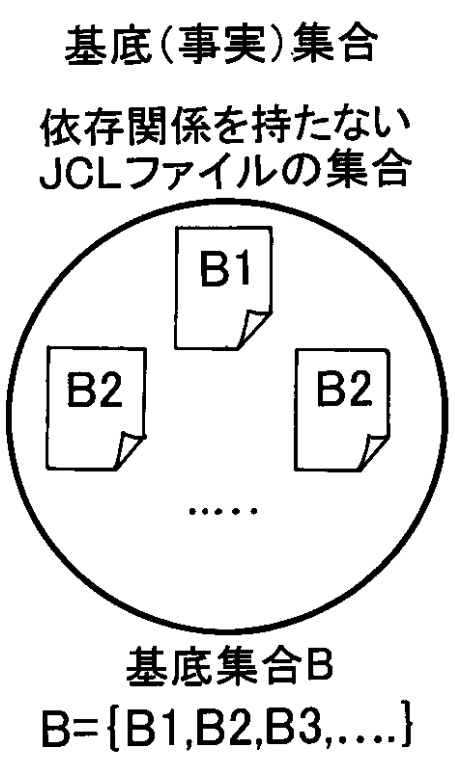
【図４】


依存関係解決済
中間ファイル集合
113-1
中間ファイル入力部
113-3
中間ファイル解析部
113-5
ジョブネットワーク
出力部
ジョブネットワーク集合

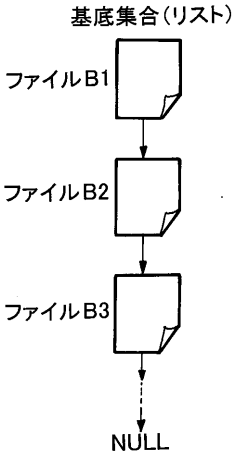
【図５】


JCLファイル集合
解析開始
S201
JCLファイル集合の
大きさが1以上
YES
NO
S203
JCLファイルを
1つPick up
S205
JCLファイルを
オープン
JCLファイル集合
解析終了
S207
JCLファイルを
read LINE
S209
EOFではない？
YES
NO
S211
依存関係指定行
であるか？
YES
NO
S213
依存関係登録
S215
JCLファイル
クローズ
S217
依存関係が1以上
存在する
NO
YES
S219
JCLファイルを
論理式集合に
所属させる
S221
JCLファイルを
基底集合に
所属させる

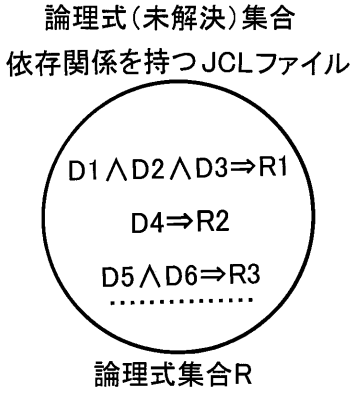
【図６】


基底(事実)集合
依存関係を持たない
JCLファイルの集合
B1
B2
B2
.....
基底集合B
B={B1,B2,B3,....}

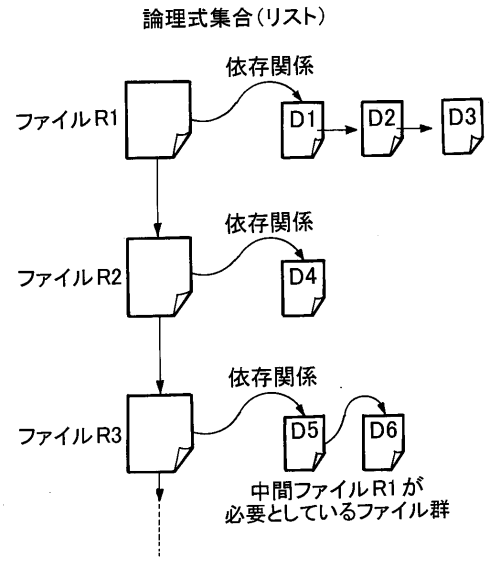
【図７】


基底集合(リスト)
ファイルB1
ファイルB2
ファイルB3
NULL

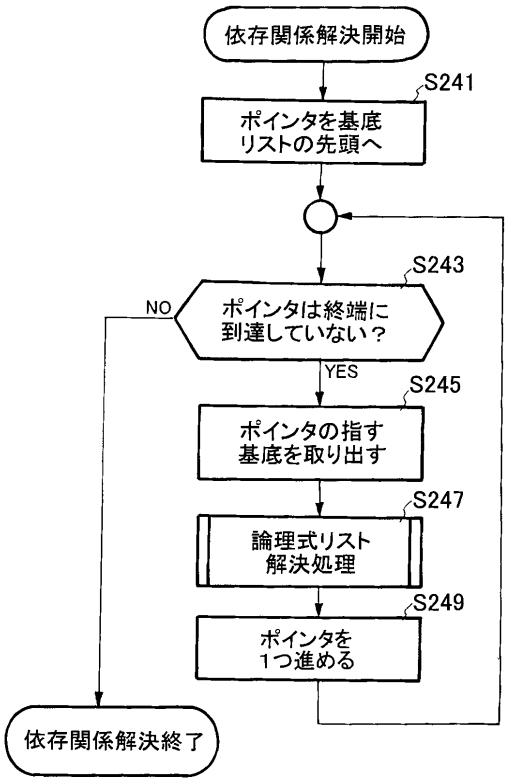
【図８】


論理式(未解決)集合
依存関係を持つJCLファイル
D1∧D2∧D3⇒R1
D4⇒R2
D5∧D6⇒R3
論理式集合R

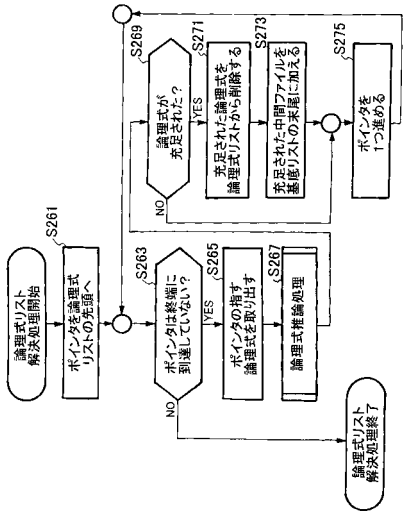
【図９】


論理式集合(リスト)
依存関係
ファイルR1
D1
D2
D3
依存関係
ファイルR2
D4
依存関係
ファイルR3
D5
D6
中間ファイルR1が
必要としているファイル群

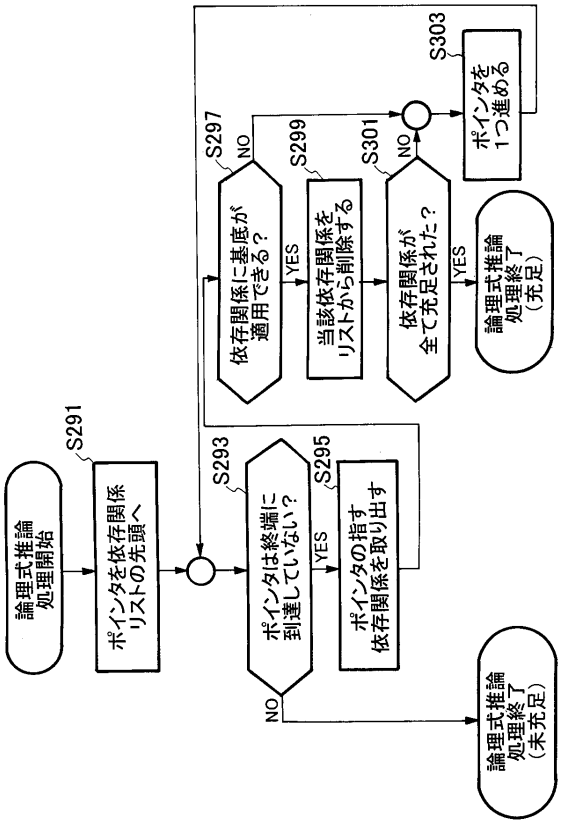
【図１０】


依存関係解決開始
S241
ポインタを基底
リストの先頭へ
S243
NO
ポインタは終端に
到達していない？
YES
S245
ポインタの指す
基底を取り出す
S247
論理式リスト
解決処理
S249
ポインタを
1つ進める
依存関係解決終了

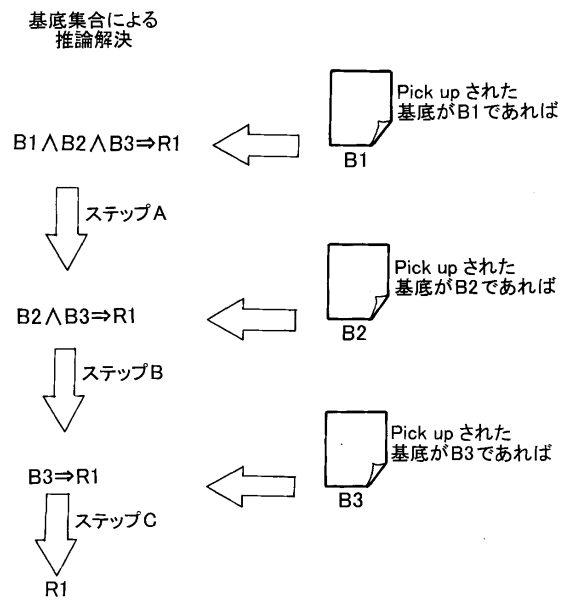
【図１１】



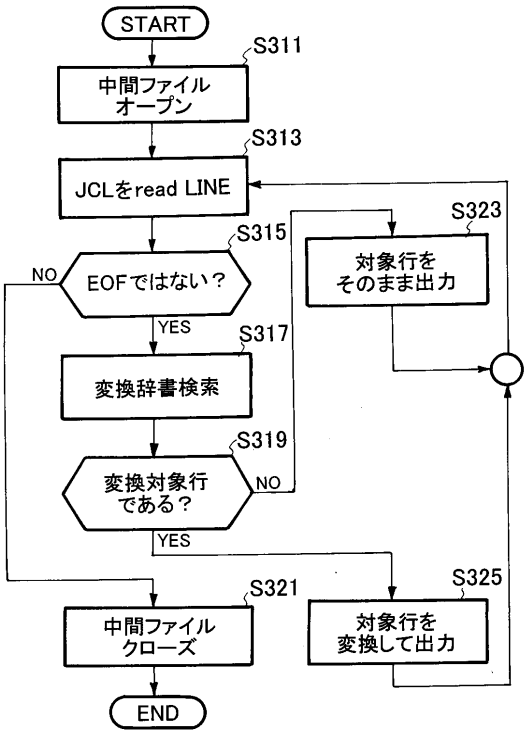
【図１２】



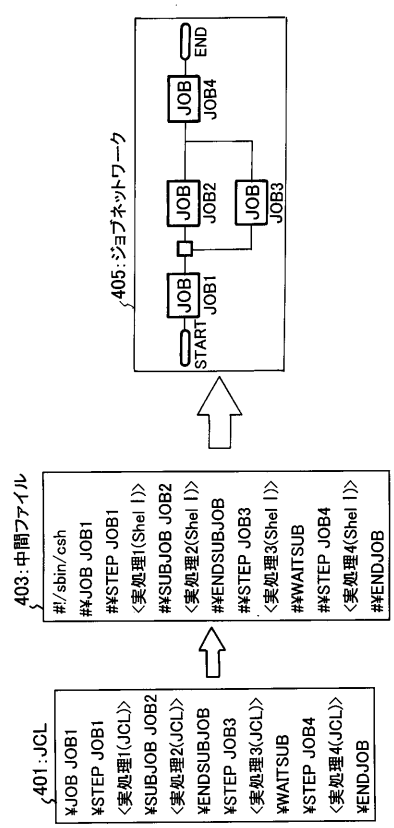
【図１３】



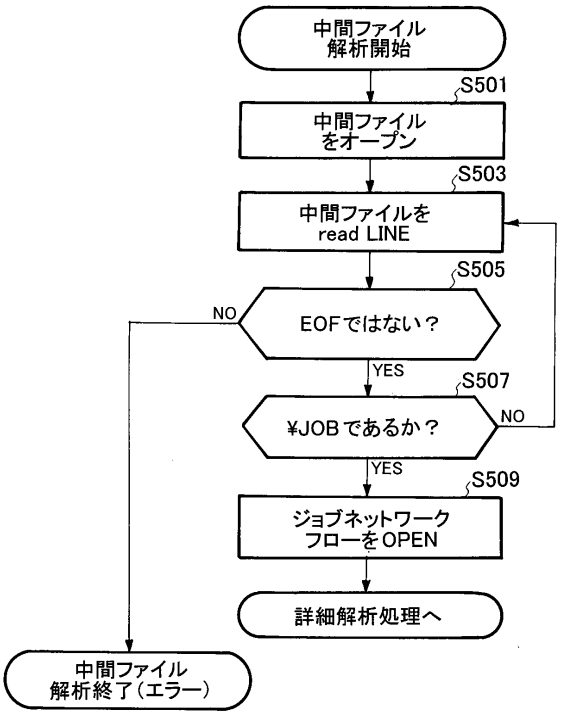
【図１４】



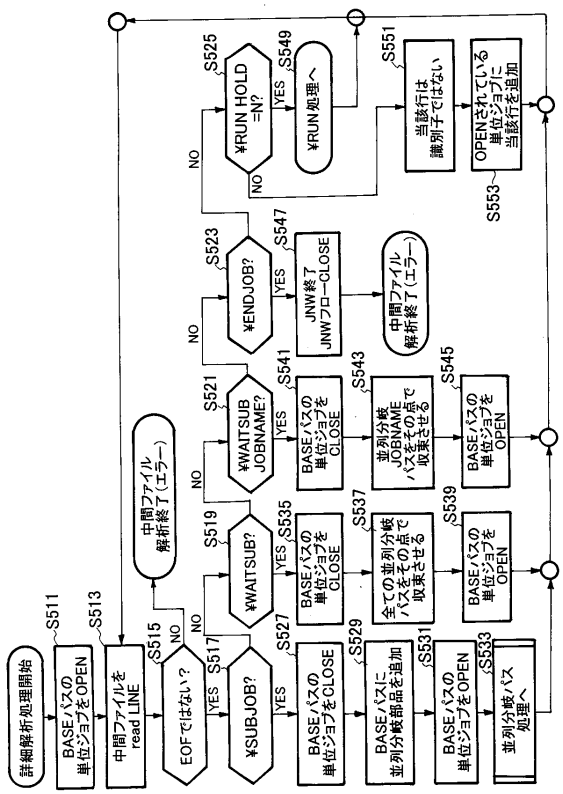
【図１５】



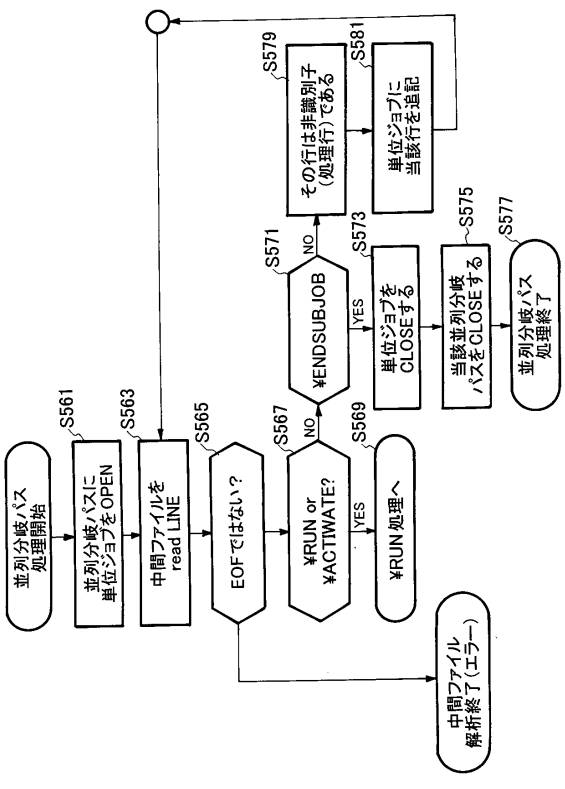
【図１６】



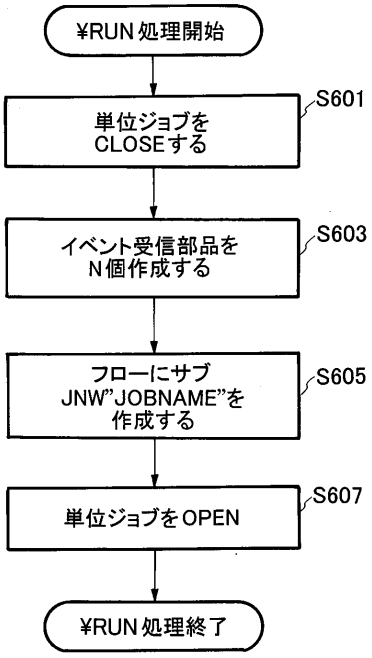
【図１７】



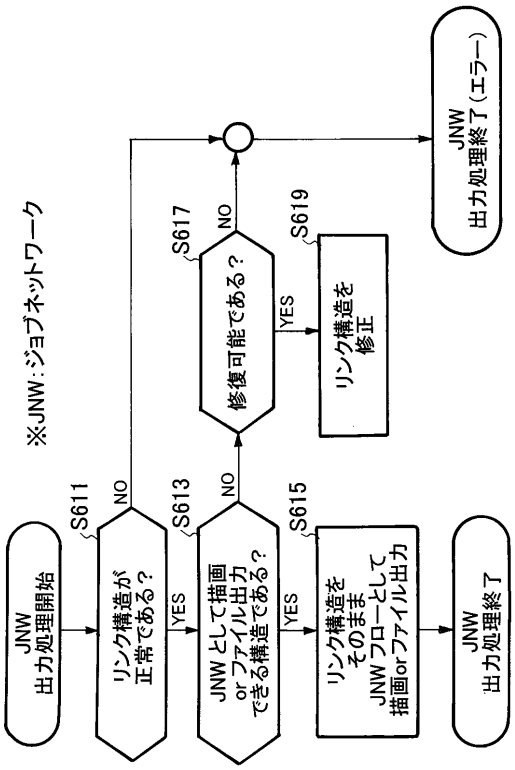
【図１８】



【図１９】


¥RUN処理開始
単位ジョブを
CLOSEする
S601
イベント受信部品を
N個作成する
S603
フローにサブ
JNW"JOBNAME"を
作成する
S605
単位ジョブをOPEN
S607
¥RUN処理終了

【図２０】


※JNW:ジョブネットワーク
JNW
出力処理開始
S611
リンク構造が
正常である？
NO
YES
S613
JNWとして描画
or ファイル出力
できる構造である？
NO
YES
S615
リンク構造を
そのまま
JNWフローとして
描画orファイル出力
JNW
出力処理終了
S617
修復可能である？
NO
YES
S619
リンク構造を
修正
JNW
出力処理終了（エラー）

フロントページの続き

合議体
審判長　赤川　誠一
審判官　田中　秀人
審判官　石井　茂和

(56)参考文献　特開平０６－０５９９１３（ＪＰ，Ａ）
特開平０１－２３７７２６（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
G06F 9/06 610W，G06F 9/06 620A