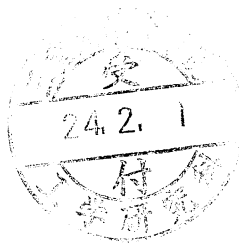


修士論文

短答記述式テストの
採点支援システムの開発
— キーワードの使われ方に
基づいた解答の分類 —



平成23年度修了

三重大学大学院 工学研究科

博士前期課程 電気電子工学専攻

大島 一真

目次

第1章	はじめに	1
第2章	テスト採点時の問題点	3
2.1	テストの解答形式	3
2.2	短答記述式テストにおける採点時の問題点	4
第3章	採点支援のためのポイント	8
3.1	各解答の採点	8
3.2	採点の公平性の維持	9
第4章	支援方法	10
4.1	基本方針	10
4.2	類似解答判定方法	11
4.3	解答の表示方法	14
第5章	採点支援システムの開発	16
第6章	実験	24
6.1	実験内容	24
6.2	実験結果	29
6.2.1	問1の実験結果	29
6.2.2	問2の実験結果	30
6.2.3	実験結果のまとめ	31
6.3	結果の考察	32
6.3.1	抜き出し終えた時間についての考察	32
6.3.2	抜き出しの妥当性についての考察	33

6.4 今後の方針	34
第7章 まとめ	35
謝辞	36
参考文献	37
発表論文	38

目 次

2.1	テストの流れ	4
2.2	採点の支援	5
2.3	円錐形 D マップ	7
4.1	注目解答の係り受け	12
4.2	類似解答判定例	13
5.1	システムの概要	16
5.2	注目解答の選択 (メインウィンドウ)	18
5.3	キーワードの入力 (サブウィンドウ)	19
5.4	共通文節数の選択 (サブウィンドウ)	20
5.5	類似解答群の表示 (サブウィンドウ)	21
5.6	類似解答群の採点 (サブウィンドウ)	22
5.7	採点結果の反映 (メインウィンドウ)	23
6.1	実験の組み合わせ	26
6.2	実験に使用した計算機の性能	26

表 目 次

4.1	強調表示例	15
6.1	問 1 の実験結果 (時間)	29
6.2	問 1 の実験結果 (内訳)	29
6.3	問 1 の実験結果 (時間)	30
6.4	問 1 の実験結果 (内訳)	30
6.5	実験結果のまとめ	31

第1章 はじめに

日本の大学における講義は、講師が受講者に対して知識を与えるだけといった、一方的なものになりがちである。講師は受講者の理解が不足している箇所に対する補足説明などができず、受講者の理解度が低いまま講義が進むといった事態になる。そこで、講師が理解状況を把握するためにテストを課すことは効果的である。また、テストでは、受講者は自分が理解したと思っていることを文章などで表現する必要がある、それによって受講者に理解の定着を促すことができる。テストを実施した後に、受講者に採点結果をすばやく返却することにより、解答が誤答であった場合に受講者は自分がどのように考えて間違いにいたったのかを思い返せるため、受講者の理解の誤りの修正を促すことができる。

テストにはさまざまな解答形式が存在し、主なものとして選択式と記述式がある。選択式のテストの解答は容易に採点できるが、受講者は講義の内容を十分に理解していなくても、当て推量で正答できる可能性がある。それに対して、記述式のテストは、受講者が自身の文章で答える必要がある、講義の内容を十分に理解していなければ正答するのが困難である。また、受講者が自身の文章で答えるため、受講者の理解を定着させるという目的のもとでテストを行うには記述式の方が適している [1]。しかし、多人数の講義では、記述式の解答は採点に時間がかかる、また、解答の内容がさまざまであるため、採点基準が揺らぎやすく公平な採点が行いにくいといった問題がある。

これをふまえて、記述式解答の採点の支援を計算機で行う試みが、多くなされている [2-8]。その中でも有名なものに、石岡らによって開発された日本語小論文の自動採点システム Jess がある [1]。Jess では内容よりは、文章の構造に重みを多くした採点を行っている。しかし、短答記述式テストの場合、文章の構造よりは内容を見て採点する必要があるため、Jess の採点手法では正しく評価できない。

本研究では、短答記述式の解答を対象とし、その採点を効率よく行うことがで

きるように、コンピュータシステムを用いて講師を支援することを目的とする。本論文では、講師が選択した解答・キーワードをもとに類似解答群を提示することを提案する。提案法によって、講師は、類似解答をまとめて採点することができる。その結果、採点基準が揺らぎにくくなり、解答の見直しが減り、採点効率を向上できる。さらに、提案法をシステムに実装し、評価実験を行うことで提案法の利点・欠点を探る。

本論文の構成を以下に示す。2章ではテストの解答形式と採点の問題点を述べる。3章で支援のためのポイントについて分析し、4章でそれをふまえた支援方法について提案し、5章で提案法に基づいて構築したシステムについて述べる。6章では実験を通じて提案法を評価し、最後に7章で本論文をまとめる。

第2章 テスト採点時の問題点

本章では、テスト採点時の問題点について述べる。2.1節ではテストの解答形式について述べる。2.2節では短答記述式テストにおける採点時の問題点について述べる。

2.1 テストの解答形式

一般に、大学などの受講者が100人にもおよぶような多人数の講義では、講師から受講者へ知識を与えるだけといった一方向講義になりがちである。その結果、講師は受講者の理解が不足している箇所に対する補足説明などができず、受講者の理解度が低いまま講義が進むといった事態になる。そこで、講師が受講者にテストを課すことは効果的である。講師は、受講者にテストを解かせ、その採点をするにより、受講者全体の理解度を解答から具体的に把握することができる。これにより講師は、受講者の理解が不足している箇所に対する補足説明を次の講義に行うなどの講義改善を適切に行うことができる。その結果、受講者の理解の誤りの修正を促すことができ、理解度が向上する。一方、受講者にとって、自分の理解した内容を文章などで表現することによって理解を定着させることができる。また、自分の解答の間違いを指摘されることによって、どのような問題でつまずき、理解が足りなかったのか知ることができ、自主的に学習するきっかけになる。

テストにはさまざまな解答形式がある。代表的な解答形式として、多肢選択式テスト、記述式テストがある。多肢選択式テストとは、問題に対する複数の選択肢のうちどれが正解であるかを、受講者が選択する形式のテストである。記述式テストとは、問題に対して受講者が自由記述で解答する形式のテストである。

本論文では、記述式テストの中でも、授業後の演習としておこなわれる短答記述式テストに着目する。本研究では200字程度以下の解答を想定している。受講

者の理解を定着させ、また、理解状況を把握するという目的で、短答記述式テストが用いられることがある [1]。その理由は、多肢選択式テストの場合、受講者が講義の内容を十分に理解していなくても、当て推量で正答できる可能性があるのに対し、短答記述式テストの場合、受講者が自身の文章で答える必要があるため、講義の内容を十分に理解していなければ正答するのが困難であるという点にある。また、講師にとって、短答記述式テストの方が採点の手間はかかってしまうが、問題を作成する手間が少ないという理由もある。多肢選択式の場合、受講者が当て推量で正答することができないようにするには、精巧な選択肢をいくつも作る必要があり、どのような選択肢を作るかを考える手間がかかる。それに対して、短答記述式テストの場合、講師は問題文とそれに対する模範解答を作成すればよく、多肢選択式よりも問題の作成に手間がかからない。以上の理由により、本論文では短答記述式テストを対象とする。

2.2 短答記述式テストにおける採点時の問題点

一般にテストの方法は、以下に示す流れで行われる (図 2.1)。

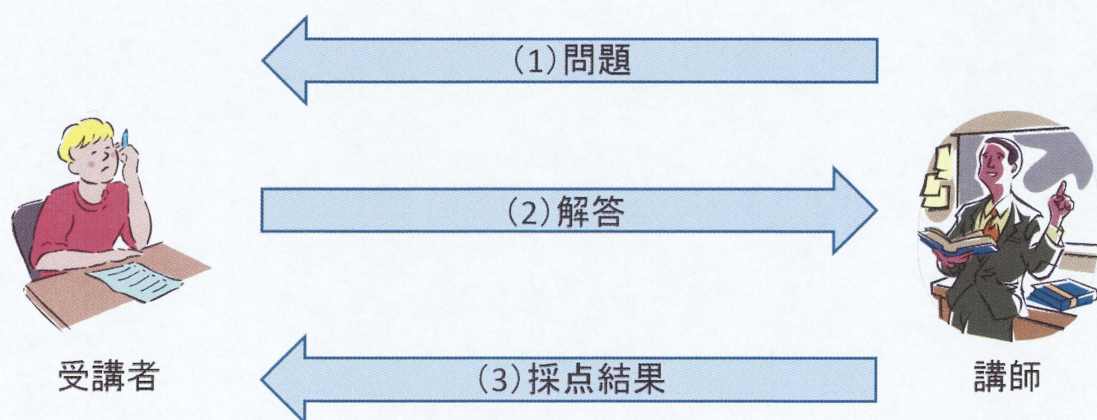


図 2.1: テストの流れ

- (1) 講師は、受講者に対して出題する。
- (2) 受講者は、解答を作成し、講師に提出する。
- (3) 講師は、解答を採点し、受講者に採点結果を返却する。

ここで、受講者にとって、採点結果はできるだけ早く返却された方が、受講者は自分の間違えた個所がどのような考え方をしたのかを覚えているため、その理解の誤りの修正を促すことができ、理解の定着を助けることができる。

小テストやレポートにおいて、短答記述式テストを用いる場合、講師は採点の公平性を保つために、類似解答を何度も読み返しながら、採点基準を維持している。しかし、一般に、大学の講義などでは受講者が100人を超えることもあり、提出された解答も多く、内容も多種多様となる。講師の負担は大幅に増加し、採点基準を維持するのが困難になる。その結果、作業効率が悪くなるといった問題が生じる。そこで本論文では、短答記述式テストの採点を効率よくできるように、コンピュータシステムを用いて講師を支援する。具体的には、図 2.2 のように、講師と受講者の間にシステムを設置することで支援を行う。

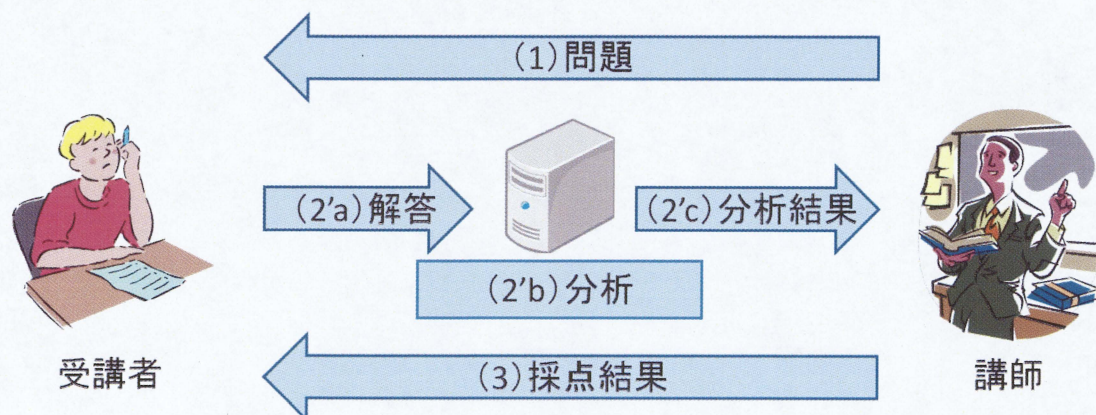


図 2.2: 採点の支援

計算機を用いて採点を支援するためのシステムについてさまざまな研究が行われてきている [2-8]. 文献 [3] で石岡らは日本語小論文の自動採点システム Jess を開発し、文献 [4] で椿本らはレポートの内容を可視化するシステムを開発した。

石岡らは、小論文の採点者の違いによる得点結果への誤差の影響を排除するためにコンピュータによる自動採点システム Jess を開発した。Jess では、(1) 修辞、(2) 論理構成、(3) 内容の三つの観点から小論文を評価し、三つの観点に係る配点を 5, 2, 3 の合計 10 点として採点を行う。ここで、修辞は漢字/カナの割合、ユールの K 特性値、ビッグワードの割合で評価し、論理構成は接続詞と指示代名詞の数で評価し、内容は問題文に対して適切な内容かどうかを Latent Semantec Indexing を用いて評価する。

椿本らは、レポート採点において、大量の文章を評価していくうちに採点者内の基準が不安定になるのを防ぐために、レポートをその内容ごとの類似・非類似度によってマップ上に可視化する円錐型 D マップを開発した。円錐形 D マップでは、レポートを使用されている単語・文章の長さによって図 2.3 のように円錐形上に配置することによって、類似した内容の解答が互いの近傍に配置されるものとなっている。

上記の二つのシステムは、ともに小論文などの文字数の多い解答を対象をしている。また、文章の内容をどのような単語が使われているかで評価している。それに対して、本論文で着目する授業後に行うような短答記述式の解答は 1, 2 文で表現され、単語の有無ではなく単語の使われ方を見て採点する必要があるため、石岡らや椿本らの手法の適用は困難である。

次章以降でこれらの問題点を解決する方法について検討する。

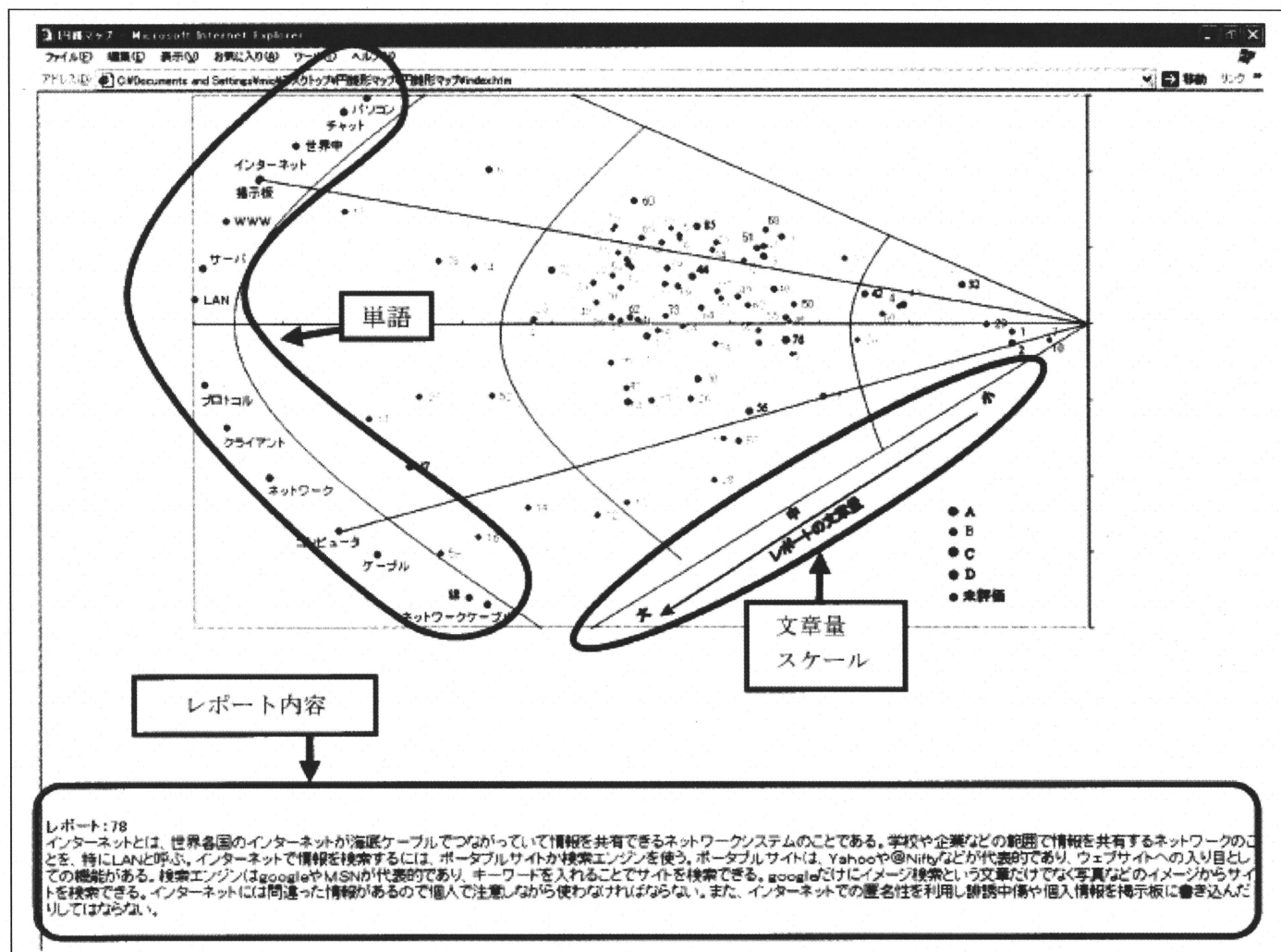


図 2.3: 円錐形 D マップ

第3章 採点支援のためのポイント

本章では、2章で述べた採点時における講師の負担をふまえ、そこから見える支援のためのポイントを活かした支援方法について議論する。記述式の採点において、講師の負担となるものは、以下の2点が大きく関係している。

1. 各解答の採点
2. 採点の公平性を保つこと

講師が短答記述式テストの採点効率を向上させるためには、これらの負担を軽減することが必要である。そこで、これらの負担から見える支援のためのポイントについて議論する。3.1節では、各解答の採点における支援のためのポイントについて検討する。3.2節では、採点の公平性を保つ際の支援のためのポイントについて検討する。

3.1 各解答の採点

実際に、講師はどのような手順で各解答を採点しているのだろうか。筆者らが本学の電気電子工学科の教員を対象に行った簡単なアンケートの結果、多くの場合、以下の手順をふんでいることがわかった。

1. キーワードが示されているかどうかを探す。
2. そのキーワードがどのように使用されているかを読む。
3. 注意を要する解答はきちんと読む。

これらの手順は解答が少数の場合、容易に行うことができる。しかし、解答が多数ある場合は、手順1のキーワードを探すのだけでも手間がかかり、手順2のキーワードの使われ方まで読むのは更に困難である。

そこで、講師が着目したキーワードの使われ方がわかりやすくなれば、手順1, 2を支援することができる考える。

3.2 採点の公平性の維持

複数の解答を採点する際には講師は類似した解答を読み比べることで同一の採点基準を維持し、公平な採点を行おうとしている。そのため、解答が多数ある場合、講師は類似した解答群についてその差異を把握することが重要である。これを把握するため、講師は、採点済みの解答も含めて全ての解答を何度も見直す。さらに、着目すべき点は多数あるため、解答が多数ある場合、その手間は膨大なものとなる。

そこで、解答が学籍番号などの順に一つずつではなく、類似解答群が示されれば、採点済みの解答を見直すことなく公平に採点できると考える。

第4章 支援方法

この章では，3章で示した各支援ポイントをふまえ，計算機により採点を支援する方法について検討する．4.1節では，支援の基本方針について述べる．4.2節では，類似解答の判定方法について述べる．4.3節では，類似解答群の表示方法について述べる．

4.1 基本方針

3章で検討したとおり，類似した解答群を講師に示すことは講師の採点の支援となる．そこで，本研究では，これを実現するインターフェースおよび類似解答の判定法を提案する．この提案に基づいた場合，講師が採点を行う手順は以下のとおりである．

- (1) 講師が，解答一覧から注目したい解答を選択する．
- (2) 講師が，注目した解答の中からキーワードを選択する．
- (3) システムが，講師によって選択された解答とキーワードの使われ方が類似している解答を抽出し提示する．

手順(1)，(2)は類似解答群を探す際に着目する「キーワードの使われ方」を指定することを意図している．キーワードを講師が指定するのは，講師や出題状況によってキーワードは異なると考えたため，キーワードは随時指定する方法が望ましいと考えられるためである．

手順(1)，(2)の実現は容易に行うことができる．そのため，次節以降で手順(3)の実現方法について述べる．

4.2 類似解答判定方法

前節で述べた方針を実現するためには、講師によって指定されたキーワードの使われ方に基づいた類似判定方法が必要となる。この節では、その方法について検討する。文書情報の解析には代表的なものとして構文解析がある。その中に係り受け文法という考えがあり、ある文節が他の文節に係る(依存する)という形式で文の構造を表現する [9]。そこで、「キーワードの使われ方」とは、キーワードがどのような前後関係で使用されているのかを意味するものと考え、言い換えると、指定されたキーワードを含む文節に係る文節と、受ける文節がどのような内容であるのかが、そのキーワードの使われ方であるとみなす。

キーワードの使われ方を表現する係り受けの関係は主に以下の2つである。

1. 主語・述語の関係
2. 修飾・被修飾の関係

例として実際に講義で出題された「狭義のコンパイラとは何ですか」という問題に対する解答と、その各文節の係り受け関係を示したものを図 4.1 に示す。

狭義のコンパイラは、高級言語で書かれたソースコードをアセンブリ言語のオブジェクトコードに変換するもの。

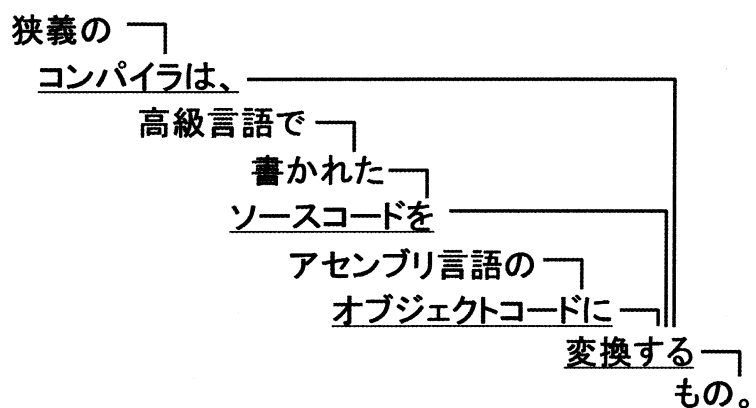


図 4.1: 注目解答の係り受け

「変換する」という述語に注目すると、これに係る文節は「コンパイラは、」, 「ソースコードを」, 「オブジェクトコードに」である。この係り受け関係によって「コンパイラは、変換する」ということがわかり、(1) 主語・述語の関係が表されていることがわかる。また、「変換する」を受ける文節は「もの」である。この係り受け関係により「ソースコードをオブジェクトコードに変換するもの」ということがわかり、(2) 修飾・被修飾の関係が表されていることがわかる。このように係り受けの関係から、「コンパイラはソースコードをオブジェクトコードに変換するもの」ということがわかる。これらはすべて「変換する」の使われ方を表している。与えられた複数の解答が類似しているかどうかは、指定したキーワードが同じ前後関係で使用されているかどうかにより判定する。具体的には、指定したキーワードを含む文節に係る文節と、その文節を受ける文節のうち、共通なものの数で判断する。すなわち、共通な文節が多いほど、指定したキーワードの使われ方が類似しているとみなす。

図 4.2 に、類似解答の判定例を示す。注目した解答が一つ目の解答で、矢印が向かう先の解答が類似しているかどうか判定される解答である。

狭義のコンパイラは、高級言語で書かれたソースコードをアセンブリ言語のオブジェクトコードに変換するもの。

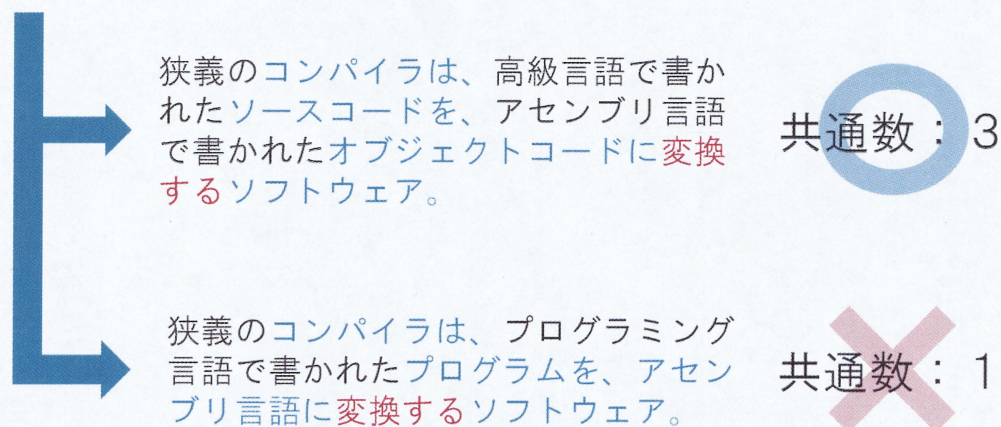


図 4.2: 類似解答判定例

ここで、「変換する」というキーワードの使われ方が類似しているかどうかを判定する。注目した解答と一つ目の解答を見ると、キーワードに係る文節と受ける文節の共通数は3である。この解答は「変換する」というキーワードを含んでいるだけでなく、「コンパイラはソースコードをオブジェクトコードに変換する」という意味を含んでいる。このように、「変換する」というキーワードの使われ方が類似した解答であることがわかる。それに対し、二つ目の解答を見ると、文節の共通数は1である。この解答は、「変換する」というキーワードは含まれているが、「コンパイラはプログラムをアセンブリ言語に変換する」という意味になり、キーワードの使われ方は類似していない。このようにキーワードに係る文節と受ける文節の共通数で解答同士の類似度を計ることができていることがわかる。

4.3 解答の表示方法

3.1 節において、講師は解答中のキーワードを見つけだし、その使われ方を見ていることがわかった。そこで、解答中の講師が注目するキーワードとその使われ方を強調して表示する。そうすることで、一目で注目したい部分を見つけることができるため、一つ一つの解答から注目する箇所を探す負担が減る。強調方法については、発表論文 [1] で示されているように、以下の三段階で表示する。

キーワードのみ

強調表示 (赤字, フォント大)

係り受け関係

強調表示 (青字, フォント中)

その他の部分

通常表示表示 (黒字, フォント小)

類似解答と判定された解答を強調表示させたものを表 4.1 に示す。このように、キーワードとその使われ方が強調表示されていることにより、講師は、類似解答群の中から注目すべき記述にすぐに目を向けることができる。

表 4.1: 強調表示例

狭義のコンパイラは、高級言語で書かれたソースコードをアセンブリ言語のオブジェクトコードに変換するもの。
狭義のコンパイラは、プリプロセッサから渡されたソースコードをアセンブリ言語に変換するもの。
狭義のコンパイラは、広義のコンパイラの中に含まれるもので、人間の作った高級言語で書かれたソースコードをアセンブリ言語などのオブジェクトコードに変換するもの。
狭義のコンパイラは、高級言語で書かれたプログラムであるソースコードを、アセンブリ言語で書かれたプログラムであるオブジェクトコードに変換するソフトウェアである。
狭義のコンパイラは、人間が読み書きできるソースコードをアセンブリ言語に変換するもの。

第5章 採点支援システムの開発

これまでの議論をふまえて、4.1節の基本方針をもとに採点支援システムを開発した。講師が使用する際の手順に従い、開発したシステムの動作を説明する。このシステムは、分析用サーバと講師側インターフェースからなり、講師は、Webブラウザを用いて、このシステムにアクセスし採点作業を行う。システムの概要図を図5.1に示す。講師側インターフェースは、Javascript および HTML で作成し、主に解答の表示を行う。類似解答判定機能は perl で作成し、分析用サーバが行う。ただし、係り受け解析は日本語構文解析システム KNP [10] を用いた。

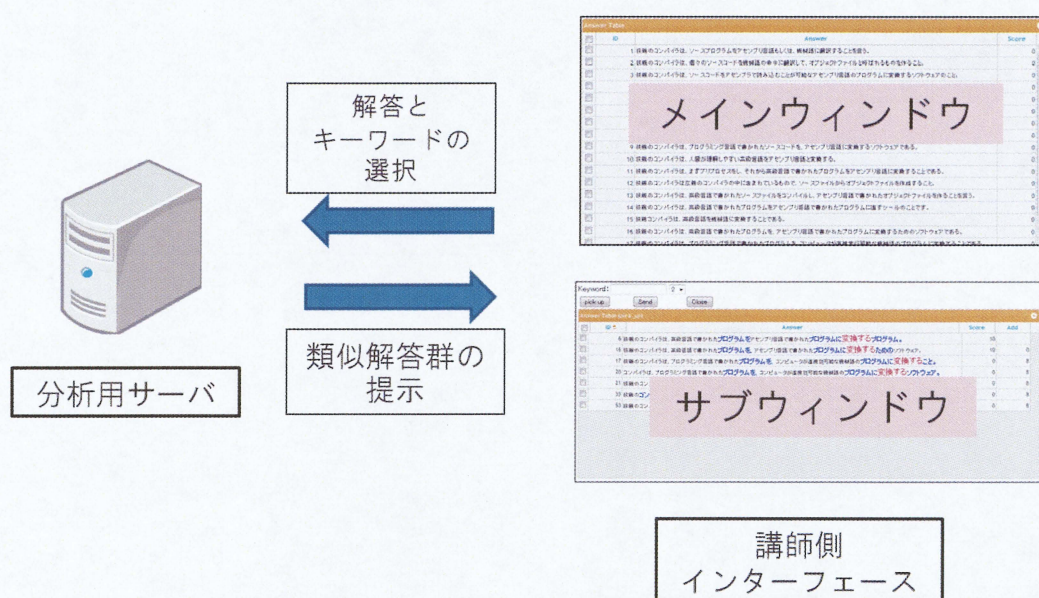


図 5.1: システムの概要

システムの使用手順を以下に示す.

- (1) 講師は, Web ブラウザからサーバにアクセスし, 全受講者の解答一覧をメインウィンドウで閲覧する.
- (2) 講師が注目したい解答をクリックで指定すると, サブウィンドウが新たに開く. 図 5.2 では, 解答一覧から ID が 7 の解答に着目し, クリックで選択している.
- (3) サブウィンドウ上で, 講師は注目するキーワードを入力する. 図 5.3 では, ウィンドウ上部の入力エリアに, キーワード「変換」を入力している.
- (4) キーワード入力エリアの右の数字を選択する. これは, 4.2 節で示した共通している文節の数の最低値を決める値であり, 小さくすればより多くの解答が類似しているとみなされ抽出され, 大きくすれば厳しく判断しているため抽出されるものは少なくなる. 図 5.4 では「2」を選択している.
- (5) システムが, 指定された解答・キーワードをもとに, その解答のキーワードの使われ方が類似している解答を 4.2 節に示した方法で抽出し, 図 5.5 のようにサブウィンドウ上に表示する.
- (6) 抽出された類似解答群を採点する. ここでは, その解答の最終的な点数を入力する方法と加点的に点数を入力する方法の 2 種類の点数入力エリアが存在する. 図 5.6 では二つの採点方法で採点している.
- (7) 採点結果を送信し, サブウィンドウを閉じると, 図 5.7 のように, メインウィンドウに採点結果が反映される.
- (8) 他の解答で (2)~(7) の動作を繰り返す.

Read

Output

Reload

ID	Answer	Score
1	扶養のコンパイラは、ソースコードをアセンブリ言語もしくは、機械語に変換すること。	0
2	扶養のコンパイラは、個々のソースコードを機械語の命令に変換して、オブジェクトコードと呼ばれるものを作ること。	0
3	扶養のコンパイラは、ソースコードをアセンブラで読み込むことが可能なアセンブリ言語のプログラムに変換するソフトウェア。	0
4	扶養のコンパイラは、人がプログラミング言語を用いて作成した設計図を、コンピューターが実行できる形式に変換するソフトウェア。	0
5	扶養のコンパイラは、プログラムソースを完全にアセンブリ言語のソースに直すだけ、もしくはそれをアセンブラする工程。	0
6	扶養のコンパイラは、高級言語で書かれたプログラムをアセンブリ言語で書かれたプログラムに変換するプログラム。	0
7	扶養のコンパイラは、高級言語で書かれたソースコードをアセンブリ言語のオブジェクトコードに変換するもの。	0
8	扶養のコンパイラは、プリプロセッサから渡されたソースコードをアセンブラという人間にわかりやすくするための機械語に変換すること。	0
9	扶養のコンパイラは、プログラミング言語で書かれたソースコードを、アセンブリ言語に変換するソフトウェア。	0
10	扶養のコンパイラは、人間が理解しやすい高級言語をアセンブリ言語に変換する。	0
11	扶養のコンパイラは、まずプリプロセッサをし、それから高級言語で書かれたプログラムをアセンブリ言語に変換すること。	0
12	扶養のコンパイラは広義のコンパイラの中に含まれているもので、ソースコードからオブジェクトコードを作成すること。	0
13	扶養のコンパイラは、高級言語で書かれたソースコードをコンパイルし、アセンブリ言語で書かれたオブジェクトコードを作ること。	0
14	扶養のコンパイラは、高級言語で書かれたプログラムをアセンブリ言語で書かれたプログラムに直すツールのこと。	0
15	扶養コンパイラは、高級言語を機械語に変換すること。	0
16	扶養のコンパイラは、高級言語で書かれたプログラムを、アセンブリ言語で書かれたプログラムに変換するためのソフトウェア。	0
17	扶養のコンパイラは、プログラミング言語で書かれたプログラムを、コンピューターが直接実行可能な機械語のプログラムに変換すること。	0
18	扶養のコンパイラは、プリプロセッサから渡されたソースコードをアセンブリ言語に変換するもの。	0
19	コンパイラは、C言語などの高級言語で書かれたソースコードを、機械語の意味を表す暗語の機械語で記述されたアセンブリ言語に変換すること。	0
20	コンパイラは、プログラミング言語で書かれたプログラムを、コンピューターが直接実行可能な機械語のプログラムに変換するソフトウェア。	0
21	扶養のコンパイラは、高級言語で書かれたプログラムをアセンブリ言語に書かれたプログラムに変換すること。	0
22	扶養のコンパイラは、広義のコンパイラの中にも含まれるもので、人間の作った高級言語で書かれたソースコードをアセンブリ言語などのオブジェクトコードに変換するもの。	0
23	扶養のコンパイラは、高級言語で作出したプログラムを、アセンブリ言語で作られたプログラムに変換するときのコンパイルを行うものをいう。	0
24	扶養のコンパイラは、高級言語で書かれたソースコードを機械語に1対1に対応したアセンブリ言語に変換すること。	0
25	扶養のコンパイラは、プリプロセッサされたプログラムをコンパイル、つまり高級言語から低級言語に変換してアセンブリ言語に置き換えるもの。	0
26	扶養のコンパイラは、人がプログラミング言語を用いて作成したソースコードを、コンピューターが実行できる形式に変換するソフトウェア。	0
27	扶養のコンパイラは、ソースコード高級言語をアセンブリ言語に変換するソフトウェア。	0
28	扶養のコンパイラは、ソースコードとして高級言語で書かれたプログラムをアセンブリ言語に変換するプログラムを持つソフトウェア。	0

図 5.2: 注目解答の選択 (メインウィンドウ)

Keyword:
2

Answer Table (Partial)				
	ID	Answer	Score	Add
<input type="checkbox"/>	7	排舞のコンパイルは、高級言語で書かれたソースコードをアセンブリ言語のオブジェクトコードに変換するもの。	0	0

図 5.3: キーワードの入力 (サブウィンドウ)

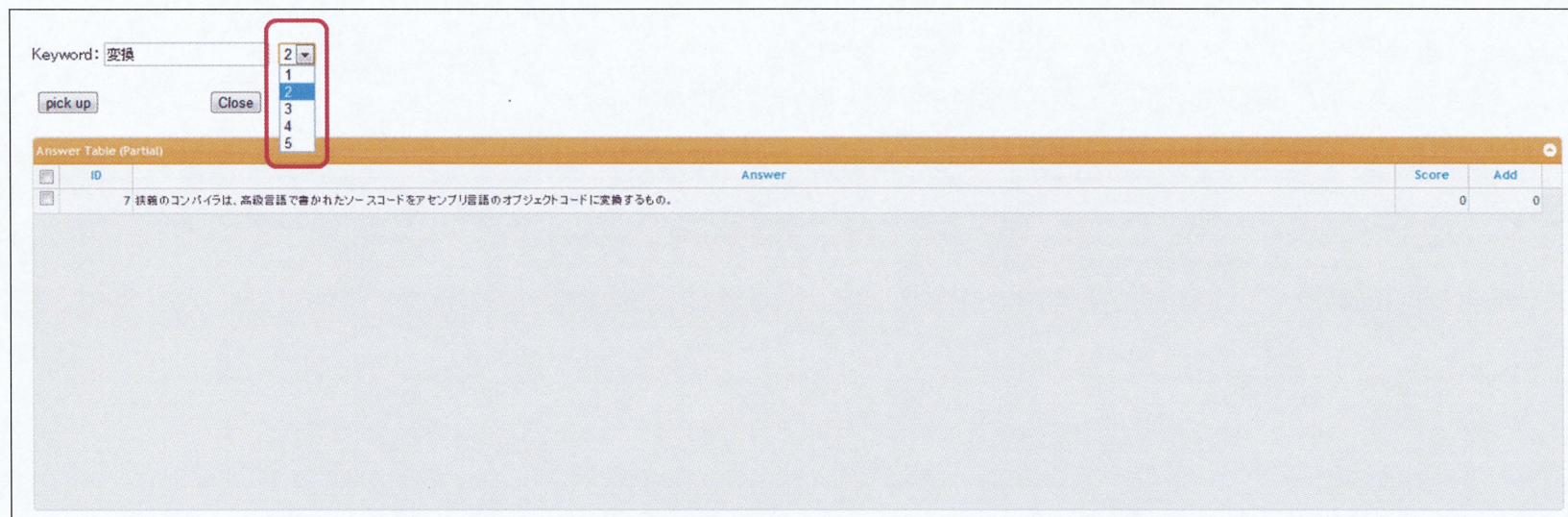


図 5.4: 共通文節数の選択 (サブウィンドウ)

Keyword: 2

pick up
Send
Close

Answer Table (pick_up)				
ID	Answer	Score	Add	
7	7 抜粋のコンパイラは、高級言語で書かれたソースコードをアセンブリ言語のオブジェクトコードに変換するもの。	0	0	
8	8 抜粋のコンパイラは、プリプロセッサから渡されたソースコードをアセンブラという人間にわかりやすくするための機械語を晦したものに変換すること。	0	0	
9	9 抜粋のコンパイラは、プログラミング言語で書かれたソースコードを、アセンブリ言語に変換するソフトウェア。	0	0	
18	18 抜粋のコンパイラは、プリプロセッサから渡されたソースコードをアセンブリ言語に変換するもの。	0	0	
19	19 コンパイラは、C言語などの高級言語で書かれたソースコードを、機械語の意味を表す暗語の機械語で記述されたアセンブリ言語に変換すること。	0	0	
22	22 抜粋のコンパイラは、広義のコンパイラの中にあまれるもので、人間の作った高級言語で書かれたソースコードをアセンブリ言語などのオブジェクトコードに変換するもの。	0	0	
27	27 抜粋のコンパイラは、ソースコードをアセンブリ言語に変換するソフトウェア。	0	0	
32	32 抜粋のコンパイラは、高級言語で書かれたプログラムであるソースコードを、アセンブリ言語で書かれたプログラムであるムオブジェクトコードに変換するソフトウェア。	0	0	
40	40 抜粋のコンパイラは、人間が読み書きできるソースコードをアセンブリ言語に変換するもの。	0	0	
43	43 抜粋のコンパイラは、C言語で書かれたソースコードをアセンブリ言語で書かれたオブジェクトコードに変換すること。	0	0	
51	51 抜粋のコンパイラは、人が書いたソースコードを、アセンブリ言語に変換するプログラム。	0	0	
52	52 抜粋のコンパイラは、プログラミング言語を用いて作成したソースコードを、コンピュータが実行できるオブジェクトコードに変換するソフトウェア。	0	0	
56	56 抜粋のコンパイラは、人間が書いたソースコードを、アセンブラが判断できる言語まで変換するという作業を行うソフトウェア。	0	0	

図 5.5: 類似解答群の表示 (サブウィンドウ)

Keyword: 2 ▼

Answer Table (pick_up)	
ID	Answer
7	扶養のコンパイルは、高級言語で書かれたソースコードをアセンブリ言語のオブジェクトコードに変換するもの。
8	扶養のコンパイルは、プリプロセッサから通されたソースコードをアセンブラという人間にわかりやすくするための機械語を晦したものに換すること。
9	扶養のコンパイルは、プログラミング言語で書かれたソースコードを、アセンブリ言語に変換するソフトウェア。
18	扶養のコンパイルは、プリプロセッサから通されたソースコードをアセンブリ言語に変換するもの。
19	コンパイルは、C言語などの高級言語で書かれたソースコードを、機械語の意味を表す略語の機械語で記述されたアセンブリ言語に変換すること。
22	扶養のコンパイルは、広義のコンパイルの中にも含まれるもので、人間の作った高級言語で書かれたソースコードをアセンブリ言語などのオブジェクトコードに変換するもの。
27	扶養のコンパイルは、ソースコードをアセンブリ言語に変換するソフトウェア。
32	扶養のコンパイルは、高級言語で書かれたプログラムであるソースコードを、アセンブリ言語で書かれたプログラムであるムオブジェクトコードに変換するソフトウェア。
40	扶養のコンパイルは、人間が読み書きできるソースコードをアセンブリ言語に変換するもの。
43	扶養のコンパイルは、C言語で書かれたソースコードをアセンブリ言語で書かれたオブジェクトコードに変換すること。
51	扶養のコンパイルは、人が書いたソースコードを、アセンブリ言語に変換するプログラム。
52	扶養のコンパイルは、プログラミング言語を用いて作成したソースコードを、コンピュータが実行できるオブジェクトコードに変換するソフトウェア。
56	扶養のコンパイルは、人間が書いたソースコードを、アセンブラが判断できる言語まで変換するという作業を行うソフトウェア。

Score	Add
5	0
0	4
0	4
0	4
0	4
5	0
0	4
5	0
0	4
0	4
0	4
5	0
0	4

図 5.6: 類似解答群の採点 (サブウィンドウ)

Read Output Reload

Answer Table			
ID	Answer	Score	
1	1 扶養のコンパイラは、ソースコードをアセンブリ言語もしくは、機械語に変換すること。	0	
2	2 扶養のコンパイラは、個々のソースコードを機械語の命令に変換して、オブジェクトコードと呼ばれるものを作ること。	0	
3	3 扶養のコンパイラは、ソースコードをアセンブラで読み込むことが可能なアセンブリ言語のプログラムに変換するソフトウェア。	0	
4	4 扶養のコンパイラは、人がプログラミング言語を用いて作成した設計図を、コンピュータが実行できる形式に変換するソフトウェア。	0	
5	5 扶養のコンパイラは、プログラムソースを単純にアセンブリ言語のソースに直すだけ、もしくはそれをアセンブラする工程。	0	
6	6 扶養のコンパイラは、高級言語で書かれたプログラムをアセンブリ言語で書かれたプログラムに変換するプログラム。	0	
7	7 扶養のコンパイラは、高級言語で書かれたソースコードをアセンブリ言語のオブジェクトコードに変換するもの。	10	
8	8 扶養のコンパイラは、プリプロセッサから渡されたソースコードをアセンブラという人間にわかりやすくなるための機械語を略したものに換すること。	8	
9	9 扶養のコンパイラは、プログラミング言語で書かれたソースコードを、アセンブリ言語に変換するソフトウェア。	8	
10	10 扶養のコンパイラは、人間が理解しやすい高級言語をアセンブリ言語と変換する。	0	
11	11 扶養のコンパイラは、まずプリプロセッサをし、それから高級言語で書かれたプログラムをアセンブリ言語に変換すること。	0	
12	12 扶養のコンパイラは応答のコンパイラの中に含まれているもので、ソースコードからオブジェクトコードを作成すること。	0	
13	13 扶養のコンパイラは、高級言語で書かれたソースコードをコンパイルし、アセンブリ言語で書かれたオブジェクトコードを作る。	0	
14	14 扶養のコンパイラは、高級言語で書かれたプログラムをアセンブリ言語で書かれたプログラムに直すツールのこと。	0	
15	15 扶養コンパイラは、高級言語を機械語に変換すること。	0	
16	16 扶養のコンパイラは、高級言語で書かれたプログラムを、アセンブリ言語で書かれたプログラムに変換するためのソフトウェア。	0	
17	17 扶養のコンパイラは、プログラミング言語で書かれたプログラムを、コンピュータが直接実行可能な機械語のプログラムに変換すること。	0	
18	18 扶養のコンパイラは、プリプロセッサから渡されたソースコードをアセンブリ言語に変換するもの。	8	
19	19 コンパイラは、C言語などの高級言語で書かれたソースコードを、機械語の意味を表す略語の機械語で記述されたアセンブリ言語に変換すること。	8	
20	20 コンパイラは、プログラミング言語で書かれたプログラムを、コンピュータが直接実行可能な機械語のプログラムに変換するソフトウェア。	0	
21	21 扶養のコンパイラは、高級言語で書かれたプログラムをアセンブリ言語に書かれたプログラムに変換すること。	0	
22	22 扶養のコンパイラは、応答のコンパイラの中にも含まれるもので、人間の作った高級言語で書かれたソースコードをアセンブリ言語などのオブジェクトコードに変換するもの。	10	
23	23 扶養のコンパイラは、高級言語で作出したプログラムを、アセンブリ言語で作られたプログラムに変換するときのコンパイルを行うものをいう。	0	
24	24 扶養のコンパイラは、高級言語で書かれたソースコードを機械語に1対1に対応したアセンブリ言語に変換すること。	0	
25	25 扶養のコンパイラは、プリプロセッサされたプログラムをコンパイル、つまり高級言語から低級言語に変換してアセンブリ言語に置き換えるもの。	0	
26	26 扶養のコンパイラは、人がプログラミング言語を用いて作成したソースコードを、コンピュータが実行できる形式に変換するソフトウェア。	0	
27	27 扶養のコンパイラは、ソースコード高級言語をアセンブリ言語に変換するソフトウェア。	8	
28	28 扶養のコンパイラは、ソースコードとして高級言語で書かれたプログラムをアセンブリ言語に変換するプログラムを持つソフトウェア。	0	

図 5.7: 採点結果の反映 (メインウィンドウ)

第6章 実験

本章では，提案したシステムによって採点基準が揺らぐことを防ぎ，採点効率を向上させることができるかどうかを実験によって調べる．6.1 節では，実験の内容について述べる．6.2 節では，実験の結果を示す．6.3 節では，実験の結果をもとに，提案法の利点・欠点について考察する．6.4 節では，考察をもとに今後，提案システムの改良すべき点について検討する．

6.1 実験内容

本実験の目的は，提案システムによって採点基準が揺らぐことを防ぎ，採点効率を向上させることができるかどうかを調べることである．そのために被験者が仮の講師役となり，提示された多数の解答から特定のキーワードの使い方をしていいる解答を選び出す時間を測定する．これにより，類似した解答を探し出す手間を評価する．短時間で正しい解答を選び出せば効率よく採点できるといえる．

具体的な評価方法は以下のとおりである．まずキーワードの使われ方に基づいて3つのクラスタとその他の解答からなる30個の解答を用意した．被験者には，各クラスタの代表となる3つの解答を示し，その解答が含まれているクラスタの中の解答を抜き出し終えた時間と，抜き出された解答の妥当性を従来システムと提案システムで計測する．ここで，従来システム・提案システムとは次に示す方法である．

従来システム 解答を解答者の ID 順に並べて提示するシステム．

提案システム 従来法に加え，類似解答を抽出できるシステム．

具体的な手順は、まず、被験者に対して従来システムで解答を提供する。次に、簡単な解答例でシステムを試用してもらい、提案システムの使い方を覚えてもらう。最後に、別の解答群を提案システムで提示する。なお、1問分の手順は以下のとおりである。

- (1) 被験者に対して、出題内容・3つの注目する解答のキーワードとその使われ方・被験者の作業内容を提示する。
- (2) 被験者は提示された解答群から、3つの解答とキーワードの使われ方が同じ解答を抜き出して用紙に記入する。
- (3) 被験者が終了を申告した時点で終了する。

被験者は、三重大大学の学部4年生および博士前期課程学生、特に計算機を日常的に扱う研究室から計12名を選んだ。また、用いた解答は、2008年度に三重大大学工学部電気電子工学科で開講された「計算機基礎Ⅱおよび演習」において実施したテストの解答2問分である。なお、本実験では全解答中の類似語はある程度統一している。被験者への解答の提示方法は表6.1のようにした。この表は各被験者が、各問をどの順序・どの手法で実験するかを示したものであり、被験者6名を1セットとし、2セット分をおこなった。また、実験に使用した計算機の性能を表6.2に示す。

受講者に提示した実験用紙を以下に示す。

図 6.1: 実験の組み合わせ

問	被験者	
	a	b
1	1(C)	2(P)
2	2(P)	1(C)

※ 数字は実施順, () 内は解答の提供方法を表す.
P: 提案システム C: 従来システム

図 6.2: 実験に使用した計算機の性能

	OS	CPU	クロック数	メモリ容量
サーバ	FreeBSD 8.2	core 2 Duo	2.66GHz	1.00GB
インターフェース	windows7	Core i3	2.93GHz	3.18GB

<div style="border: 1px solid black; padding: 5px; margin-bottom: 10px;"> 問題: 狭義のコンパイラとは何か </div> <p style="text-align: center; margin-bottom: 10px;"> 注目する解答と キーワードの使われ方 </p> <table border="1" style="width: 100%; border-collapse: collapse; margin-bottom: 10px;"> <thead> <tr> <th style="width: 10%;">解答番号</th> <th style="width: 90%;">解答</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">1</td> <td>ソースコードをアセンブリ言語に変換する</td> </tr> <tr> <td style="text-align: center;">2</td> <td>ソースコードをオブジェクトコードに変換する</td> </tr> <tr> <td style="text-align: center;">3</td> <td>ソースコードを機械語に変換する</td> </tr> </tbody> </table> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 10%;">注目する 解答の番号</th> <th colspan="12">選択する解答の番号</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">1</td> <td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td> </tr> <tr> <td style="text-align: center;">2</td> <td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td> </tr> <tr> <td style="text-align: center;">3</td> <td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td> </tr> </tbody> </table>													解答番号	解答	1	ソースコードをアセンブリ言語に 変換 する	2	ソースコードをオブジェクトコードに 変換 する	3	ソースコードを機械語に 変換 する	注目する 解答の番号	選択する解答の番号												1													2													3												
解答番号	解答																																																																							
1	ソースコードをアセンブリ言語に 変換 する																																																																							
2	ソースコードをオブジェクトコードに 変換 する																																																																							
3	ソースコードを機械語に 変換 する																																																																							
注目する 解答の番号	選択する解答の番号																																																																							
1																																																																								
2																																																																								
3																																																																								

図6.1: 実験用紙1 (狭義のコンパイラとは何か)

<div style="border: 1px solid black; display: inline-block; padding: 5px; margin-bottom: 10px;">問題: 機械語とは何か</div> <p style="margin-top: 20px;">注目する解答と キーワードの使われ方</p>												
解答番号	解答											
1	直接実行できる 二進数 で書かれた言語											
2	二進数 に変換する											
3	0と1の 二進数 で表現される											

注目する 解答の番号	選択する解答の番号											
1												
2												
3												

図6.2: 実験用紙2 (機械語とは何か)

6.2 実験結果

本節では実験結果を示す。問 1,2 に対する実験結果をそれぞれ 6.2.1 節, 6.2.2 節で示す。

6.2.1 問 1 の実験結果

問 1 の実験結果を表 6.1, 表 6.2 に示す。表 6.1 では抜き出し終えた時間, 表 6.2 では抜き出した解答の内訳である。

表 6.1: 問 1 の実験結果 (時間)

	従来システム	提案システム
平均 [s]	299	270

表 6.2: 問 1 の実験結果 (内訳)

	従来システム				提案システム			
	選択数	正	不足	誤	選択数	正	不足	誤
平均 [個]	16.7	12.8	2.8	3.8	11.8	11.3	4.5	0.5

選択数：選択された解答数 正：正しく選択された解答数

不足：選択から漏れた解答数 誤：誤って選択された解答数

まず、被験者が解答の抜き出しが終了するまでの時間を見る。表 6.1 より、従来システム・提案システムにおいて、被験者が解答の抜き出しを終了するまでの時間の平均は 299 秒, 270 秒であった。t 検定の結果, 5%水準で有意差は見られなかった。

次に被験者が抜き出した解答の妥当性を見る。表 6.2 より、従来システム・提案システムを用いて被験者が抜き出すことができなかった解答数の平均はそれぞれ 2.8 個, 4.5 個である。また、誤って抜き出された解答数の平均はそれぞれ 3.8 個, 0.5 個である。t 検定の結果, 5%水準で選択から漏れた解答数には有意差傾向 ($0.05 < p < 0.10$) が見られ、誤って抜き出された解答数には有意差があった。

6.2.2 問2の実験結果

実験結果を表 6.3, 表 6.4 に示す.

表 6.3: 問1の実験結果 (時間)

	従来システム	提案システム
平均 [s]	363	157

表 6.4: 問1の実験結果 (内訳)

	従来システム				提案システム			
	選択数	正	不足	誤	選択数	正	不足	誤
平均 [個]	21.1	9.5	0.5	11.7	10.2	9.3	0.7	0.8

選択数：選択された解答数 正：正しく選択された解答数

不足：選択から漏れた解答数 誤：誤って選択された解答数

前節と同様に，被験者が解答の抜き出しが終了するまでの時間を見る．表 6.3 より，従来システム・提案システムにおいて，被験者が解答の抜き出しを終了するまでの時間の平均は 363 秒，157 秒であった． t 検定の結果，5%水準で有意差が見られた．

次に被験者が抜き出した解答の妥当性を見る．表 6.4 より，従来システム・提案システムにおいて抜き出すことができなかった解答数の平均はそれぞれ 0.5 個，0.7 個である．また，誤って抜き出された解答数の平均はそれぞれ 11.7 個，0.8 個である． t 検定の結果，5%水準で選択から漏れた解答数には有意差が見られず，誤って抜き出された解答数には有意差が見られた．

6.2.3 実験結果のまとめ

まず，表 6.5 に実験結果のまとめを示す．

表 6.5: 実験結果のまとめ

問	抜き出し終えた時間	抜き出しの妥当性	
		不足	誤
1	$P = C$ (33 秒)	$P = C$	$P > C$
2	$P > C$ (149 秒)	$P = C$	$P > C$

不足：選択から漏れた解答数 誤：誤って選択された解答数

P ：提案システム C ：従来システム

表 6.5 は，被験者が解答を抜き出すまでの早さと，その抜き出しが妥当かどうかを示したものである．なお，従来システムと提案システムに有意差が見られた場合は不等号を，有意差が見られなかった場合は等号を示す．また括弧内は従来システムと提案システムの抜き出すまでの時間の差である．表 6.5 により，問 1 では，両システム間で抜き出し終えた平均時間に差はなかった．しかし，抜き出しの妥当性は提案システムの方が高かった．問 2 では，提案システムの方が抜き出し終えた平均時間が短く，抜き出しの妥当性も高かった．以上より，抜き出し終える時間は提案システムの方が速いときもあれば差がない場合もあることがわかる．また，提案システムの方が抜き出しの妥当性は高いことがわかる．この結果から，提案システムを利用することにより，類似解答を見つけだすまでの時間を短縮でき，抜き出しの妥当性も高めることができる可能性が示された．

6.3 結果の考察

本節では、6.2節の結果について考察する。具体的には、採点基準が揺らぐことを防ぎ、採点効率を向上させることに対する提案システムの有効性について考察する。6.3.1節では、抜き出し終えた時間について考察する。6.3.2節では、抜き出しの妥当性について考察する。

6.3.1 抜き出し終えた時間についての考察

まず、抜き出し終えた時間について考察する。問1では従来システムと提案システム間のt検定の結果、有意差が見られなかった。つまり、両システム間に若干の差ができたが効率が向上したとは言えない。同様に、問2におけるt検定の結果、有意差が見られた。つまり、両システム間に差ができており効率が向上したといえる。

次に両システム間に問2では有意差が見られたが、問1では有意差が見られなかった理由について考察する。問によって両システム間の有意差に違いが出た理由は、被験者への問題の設定にその違いの理由があるのではないかと考える。被験者へ提示したキーワードは、問1では「変換」、問2では「二進数」であった。「変換」は「変換する」という動詞に含まれている単語で、「二進数」は助詞が付く単語であり、これらに係り受けする文節の数は「変換」という動詞に含まれる単語をキーワードとした方が多くなっている。そのため、提案システムにおいて、5章のシステム使用手順(4)で示した共通の文節の最低値を小さくすると、抜き出される解答数は多くなってしまう。そこで、さらに解答数を絞るために最低値を大きくしていくといった動作が増える。つまり、キーワードとする単語によって、最適な最低値を見つけだすまでの時間が余分にかかってしまい、従来システムとの差が小さくなったと考えられる。しかし、今回の実験では解答数30、クラスタ数3としたが、実際の解答数は80程度あり、クラスタ数もより多くなる。そのため、最適な最低値を見つけだすまでの時間が余分にかかっても、従来システムのように何度も読み返すことがないため、差が大きくなる可能性はある。

6.3.2 抜き出しの妥当性についての考察

次に抜き出しの妥当性について考察する。問1について表6.2より、従来システム・提案システム間のt検定の結果、選択から漏れた解答数には有意差が見られず、誤って抜き出された解答数には有意差が見られた。同様に、問2について表6.4より、従来システム・提案システム間のt検定の結果、選択から漏れた解答数には有意差が見られず、誤って抜き出された解答数には有意差が見られた。以上の結果から、提案システムを用いたことにより誤って抜き出された解答数が少ないため採点基準が揺らぎにくくなる。しかし、抜き出すことができなかった解答数も増えてしまう可能性がある。

これまでの議論から、提案システムを用いると、誤って抜き出すことは減るが、必要な解答を抜き出すことも減ってしまうことがわかった。次にこの理由について考察する。まず、誤った抜き出しを減らすことができた理由は、提案システムを用いることにより、指定するキーワードの使われ方が類似している解答群が提示されるため、他のクラスターの解答や関係のない解答が選択肢の中に含まれることがないためであると考えられる。次に、問1において必要な解答を抜き出すことができなかった理由は、6.3.1節と同様に、被験者へ提示したキーワードが関係しているのではないかと考えた。今回の実験では、問1の「変換」という動詞に含まれている単語に係る文節の中に「ソースコードを、」というものがあつた。本研究において開発したシステムでは、係り受けする文節の一致判定には句読点の有無が考慮されていたため、「ソースコードを、」と「ソースコードを」という文節は違う文節であると判定されてしまった。そのため、提案システムにおいて、5章のシステム使用手順(4)で示した共通の文節の最低値が小さいうちは、他の係り受けしている文節が一致しているものがあつたため、類似解答群に含まれていた。しかし、共通の文節の最低値を大きくするうちに、共通の文節数が減り、類似解答群から漏れてしまった。つまり、講師により選択された基準の解答によっては、句読点の有無が悪い意味で考慮されてしまい、抽出漏れを起こしてしまい、従来システムより、必要な解答を抜き出すことができなかったと考えられる。実際に、最低値を大きくして抽出漏れしてしまった解答を見ると、句読点の有無だけで類似していないと判定されているため、句読点の考慮をしなければ抽出できた解答群であることがわかった。

6.4 今後の方針

本節では、6.3節をもとに、今後どのように提案システムを改良するかについて検討する。改良すべき点として次に示す問題点が挙げられる。

- (1) 指定するキーワードによって最適な共通の文節の最低値を見つけだすまでの時間がかかる。
- (2) 一致判定に句読点の有無を考慮したことにより、抽出漏れを起こしてしまう。

まず、問題点(1)を解決するための改良案を考える。提案システムにおいて、指定するキーワードは講師によって任意に選択されるものであるため、システムで自動的に最適な最低値を見つけだすことは困難である。そこで、最低値毎に抽出される解答数を講師に提示してやることにより、最適な最低値を見つけだす手助けを行う。これにより、講師は最適な最低値を見つけだす際に、手掛かりとなるものが提示されているため、試行錯誤を繰り返すより早く最低値を見つけだすことができると考えられる。

次に、問題点(2)を解決するための改良案を考える。この問題点は、句読点の有無によって類似していないと判定されていたことがわかっているため、単純に提案システムにおける一致判定に句読点を考慮しないようにすることで解決することができる。また、本実験では、類似語を統一したうえで実験を行った。それを行わない場合は、類似解答をうまく抽出できない恐れがある。そこで、シソーラス辞書を組み込むなど、類似語を判定できるようにすることで、句読点の有無に関わらず類似解答の抽出精度を向上させることができると考えられる。

第7章 まとめ

本論文では、短答記述式の解答を対象とし、その採点を効率よく行うことができるように、コンピュータシステムを用いて講師を支援することを目的とした。講師は解答を見る際にキーワードの使われ方に注目し、類似解答を読み比べることにより採点していることがわかった。そこで、講師が選択した解答・キーワードをもとに類似解答群を提示することを提案した。解答同士が類似しているかどうかは、講師が指定したキーワードに係る文節と、受ける文節のうち、共通なもの数で判定する。そして提案システムを構築し、システムの評価実験を通じて提案法の有効性を検証した。その結果、提案法により、類似解答を見つけだす時間を短縮することができたが、抽出のしきい値を低くした時、選択するキーワードにより多くの類似解答群が抽出されてしまい、最適なしきい値を見つけだすまでの時間が余分にかかってしまうことがわかった。そのため、この問題を解決するための改良案をいくつか示した。今後の課題として、改良案が有効に働くかどうかを検証していくことが挙げられる。

謝辞

本論文は、筆者が三重大学大学院工学研究科博士前期課程に在学中に行った研究をまとめたものである。本研究を進めるにあたり、懇切丁寧なご指導とご督励を賜った三重大学鶴岡信治教授，高瀬治彦准教授，北英彦准教授，川中普晴助教に深く感謝いたします。また，日頃熱心に討論して頂いた情報処理講座の皆様方にお礼申し上げます。

最後に，本論文をまとめるにあたり，助言，討論，その他お世話になったすべての方々に感謝いたします。

参考文献

- [1] 植野真臣, 永岡慶三, e テスティング, 培風館, 2009
- [2] 植野真臣, 荘島宏二郎, 学習評価の新潮流, 朝倉書店, 2010
- [3] 石岡恒憲, 亀田雅之, コンピュータによる小論文の自動採点システム Jess の試作, 計算機統計学, Vol. 16, No. 1, pp. 3-19, 2003
- [4] 椿本弥生, 柳沢昌義, 赤堀侃司, レポート内容とその評価を可視化する円錐形レポート採点支援マップの開発と評価, 日本教育工学会論文誌, Vol. 31, No. 3, pp. 317-326, 2007
- [5] 村田淳哉, 片上大輔, 新田克己, SVM を利用した小論文の採点支援システム, 電子情報通信学会技術研究報告, 人工知能と知識処理 Vol.107, No.428, pp.7-12, 2008
- [6] 石川慧, 藤井紀子, 藤井誠, 小濱隆司, 採点支援システムの研究・開発, 電子情報通信学会総合大会講演論文集 2007 年 情報・システム (1), p.165, 2007
- [7] 中村泰介, 高瀬治彦, 森田直樹, 川中普晴, 鶴岡信治, 短答式記述テストの支援システム - 作成途中の解答の表示方法に関する検討 -, 2010 PC Conference 論文集, pp.239-242, 2010
- [8] 野呂和誉, 記述式小テストの解答の傾向を即時に把握するシステムの構築, 三重大学大学院工学研究科修士論文, 2007
- [9] 長尾真, 自然言語処理, 岩波書店, 1996
- [10] 日本語構文解析システム KNP,
<http://nlp.ist.i.kyoto-u.ac.jp/index.php?日本語構文解析システム KNP>

発表論文

- [1] 大島一真, 中村泰介, 高瀬治彦, 川中普晴, 鶴岡信治, 短答記述式テストの支援システム — 解答のキーワード強調に関する検討 —, 情報学ワークショップ 2010(WiNF2010) 論文集, pp. 231–234, 2010
- [2] 大島一真, 高瀬治彦, 川中普晴, 鶴岡信治, 短答記述式テストの採点支援システムの開発 — キーワードの使われ方に基づいた解答の分類 —, 2011 PC Conference 論文集, pp. 18–21, 2011
- [3] Kazuma Oshima, Haruhiko Takase, Hiroharu Kawanaka, Shinji Tsuruoka, Development of Descriptive Short-answer Test Scoring Support System — Answer Clustering Based on Usage of Keyword —, International Symposium for Sustainability by Engineering at MIU 2011, pp. 265–266, 2011