社団法人　情報処理学会　研究報告
IPSJ SIG Technical Report

2006－CE－83（18）
2006／2／18

プログラミング演習支援システムCAPESのための答案評価機構の実現

中島秀樹† 宮地恵佑† 高橋直久†

本稿では，プログラミング演習支援システムCAPESのための答案評価機構を提案し，プロトタイプを用いて評価する．提案機構は答案プログラムの実行誤りによるCAPESの破壊を防ぐ機能，教師によって指定された実行環境で答案プログラムを実行する機能，関数や変数の値を詳細に調べる機能，複数のモジュールからなるプログラムを部分的に評価する機能を有する．これらにより，多様な課題に対応した評価が可能になる．

Learner's Program Evaluation Mechanism for Computer Aided Programming Exercise System (CAPES)

Hideki Nakajima,† Keisuke Miyachi† and Naohisa Takahashi †

We propose a learner's program evaluation mechanism for a computer aided programming exercise system (CAPES) and apply the prototype to actual classes. CAPES includes the four following functions: (1) to execute a learner's program safely, (2) to execute a learner's program by using a virtual machine, (3) to trace values of functions and variables, and (4) to execute a portion of a learner's program. These functions enable CAPES to evaluate a variety of programming exercises safely.

1. はじめに

筆者らはこれまでに，受講者の習得度に応じた手続き型プログラミング演習のためのQAサイクル実行システムCAPES（Computer Aided Programming Exercise System[1)]）の研究を行い，CAPESを実際のプログラミング講義（Pascal，C）の演習に適用してきた．ここで，QAサイクルとは，受講者への問題の提示から答案プログラムの評価までの一連の流れである．本稿では，教師が受講者に理解させたい機能（手続き）や，習得させたい技能を課題と呼び，課題をより具体的な形で受講者に提示するものを問題と呼び区別する．1つの課題に対して複数の問題を繰り返し解くことにより，課題に対する習得度を高めるようにする．CAPESは，受講者にプログラミング技能を身に付けさせるために，受講者毎に違う問題を与えたり，問題の難易度を受講者の習得度に応じて半自動的に変更しながら，受講者がQAサイクルを行うことを支援するものである．CAPESでは，プログラミング演習のQAサイクルを記述するための言語EDML（Exercise Design Markup Language）を用いて教師により記述されたQAサイクルを半自動的に実行する．ここでEDMLとは，答案評価条件（正解プログラム，実行コマンド，入力データ等）等を記述する言語である．習得度の点数化については，手続き型プログラムが手続き間の値の受け渡しを変数への代入操作によって行うことに着目し，答案プログラム実行時の各手続きの結果として得られる変数の妥当性を評価することにより行う．これより，手続き型プログラムのどの手続きまで理解し，習得できたのかを判断する．

本稿では，次の4つの特徴をもつ答案評価機構を提案する．これらの特徴により，CAPESが多様な課題に対応することが可能になる．

特徴1　安全な答案プログラム実行　安全に答案プログラムを実行するために，EDMLで記述されたプログラムの実行環境から答案プログラムの実行に必要な資源を求めて，仮想環境（サンドボックス[2)]）を構築する機能を実現する[3)]．また，仮想機械や仮想OSのようにハードウェアやOS全体を仮想化するのではなく，資源の情報のみを仮想化してオーバヘッドの小さいサンドボックスを

† 名古屋工業大学
Nagoya Institute of Technology

構築する機能を実現する．これにより，ファイルを操作するような課題において，受講者の予期せぬプログラムの実行による，CAPES にとって大切なファイルの変更・削除を防ぐことが可能となる．また，扱う資源が異なる演習において，QA サイクルの応答性をほとんど劣化させることなく，EDML 文書を設計するだけで毎回自動でサンドボックスを構築することが可能となる．

特徴 2　変数・関数の値の調査　変数・関数のプログラム中の実行結果（トレースデータ）を取得するために，EDML で記述された教師が評価したい変数名・関数名を用い，プログラムの変数値や関数への入出力値を調査する機能を実現する．具体的には，デバッガ[4] を利用し，評価したい変数名・関数名からデバッガを動作させるコマンドを自動生成する．そして，生成したコマンドを用いてデバッガを動作させた結果からトレースデータを取得する機能を実現する．これにより，EDML 文書を設計するだけで，答案プログラムの変更をせずに変数・関数の値を自動で調べ，トレースデータを作成することが可能となる．

特徴 3　仮想機械（VM）上での答案プログラム実行　VM に対する入力と VM 上で動作する答案プログラムに対する入力のために，入力データを分割して VM に与え，VM 上で答案プログラムを実行させる機能を実現する．これにより，ネイティブコードにして動作させることができないプログラムを VM 上で実行することが可能となる．

特徴 4　答案プログラムの部分実行　プログラムを部分的に作成した段階で評価できるようにするために，教師が用意した正解プログラムの一部と受講者の答案プログラムの一部をマージする機能を実現する．これにより，受講者が部分的に作成した答案プログラムを実行可能なプログラムに変換し，受講者の答案プログラムの一部を段階的に評価することが可能となる．

本稿では，2. で CAPES について述べ，3. で提案機構の概要について述べる．次に，4. で提案機構の実現法を示し，5. で提案機構の評価を行う．最後に，6. で結果をまとめる．

2. プログラミング演習支援システム CAPES

本章では，筆者らがこれまでに研究してきたプログラミング演習支援システム CAPES について述べる．CAPES では，教師により EDML を用いて記述された QA サイクルを半自動的に実行する．EDML の記述内容をもとに，CAPES は，問題の生成から，受講者の答案プログラムの評価及び，評価結果の提示までを自動的に行うことができる．また CAPES では，EDML で記述された答案評価条件に従って答案プログラムを実行し，その結果の評価により答案を評価する．プログラミング演習時の CAPES の実行手順は図 1 のとおりである．まず，教師は CAPES の実行手順 1-6 を実行できるように，EDML を用いて QA サイクルを設計する（EDML 文書）．教師が作成した EDML 文書を CAPES に登録することで，CAPES が演習の設定を行う（実行手順 1）．EDML 文書が登録されると，CAPES は受講者に応じて問題を生成し提示する（実行手順 2，3）．受講者から答案が提出されると，答案を解析し答案評価を行う（実行手順 4，5）．そして，答案評価結果から習得度を計算し，結果を受講者に提示する（実行手順 6）．これら実行手順 2-6 までの流れが QA サイクルである．

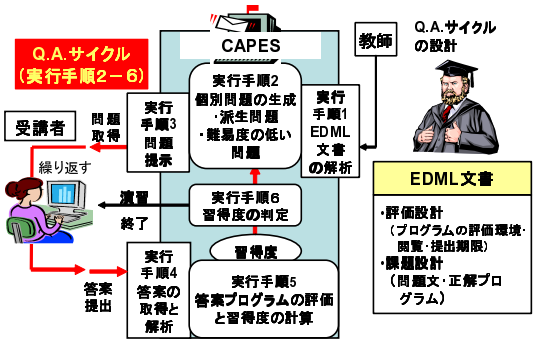

図 1　CAPES の実行手順

2.1 CAPES の答案評価機構

CAPES では，答案プログラムの変数（トレース変数）に対してトレースデータを求め，トレースデータの照合を行う機能を有する．具体的には，答案プログラムの妥当性の判定のために，教師が用意した正解プログラムと受講者の答案プログラムを，テスト技法[5] を用いた入力データ（テストデータ）を与えて実行し，その実行結果の比較を行う．テスト技法は本質的にサンプル検査であるので，答案プログラムが完全に正しいと判定することはできない．しかし，検査すべき変数とテストケースを増やすことにより，そのプログラムの実現に必要な技能をどの程度習得できたか詳細に調べることはできる．このため EDML では，複数のトレース変数や，複数のテストケースに対するテストデータを設定可能にしている．これらの記述により，答案プログラムの評価を自動化している．

CAPES の答案評価機構は図 2 の構成となっている．

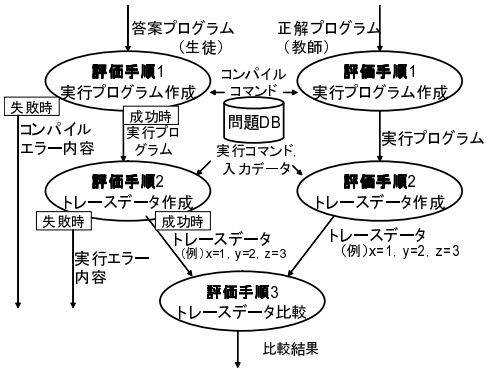

図 2　CAPES の答案評価機構

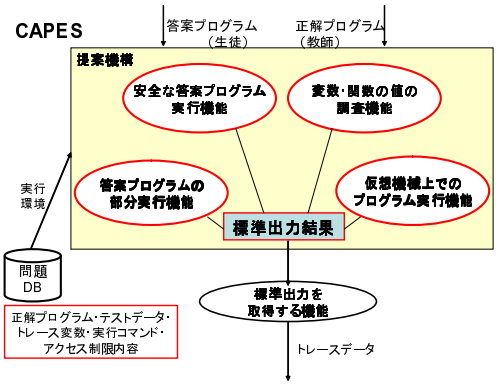

図 4　提案機構を組み込んだ答案評価機構

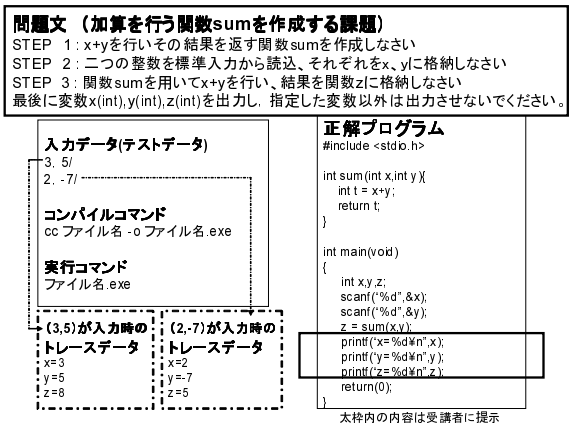

図 3　関数 sum を作成する課題の問題文と答案評価に必要なデータと出力結果の例

答案評価手順を 2 整数の加算を行う関数 sum の問題（図 3）を例にして次に説明する．ここで，図 3 の太枠部分は受講者に提示する部分である．

評価手順 1　実行プログラム作成

EDML 文書で設定したコンパイルコマンドを用いてプログラムをコンパイルする．コンパイルが失敗した時は，コンパイルエラー内容を結果として返す．図 3 の例では，EDML で記述されたコンパイルコマンド“ cc ファイル名 -o ファイル名.exe ”を用い，C プログラムをコンパイルする．

評価手順 2　トレースデータ作成

EDML 文書で設定したテストケースと実行コマンドを用いてプログラムを実行する．正常に実行できた場合，EDML で設定したトレース変数のトレースデータを取得する．実行エラー時はそのエラー内容を返す．図 3 の例では，実行コマンドとして“ ファイル名.exe ”と，入力データとして“ 3，5 ”と“ 2，7 ”を用い，プログラムを実行する．そして，トレースデータとして“ x=3 〃 y=5 ”“ z=8 ”（入力が“ 3，5 ”の時）といった 1 行毎の出力結果を取得する．

評価手順 3　トレースデータ比較

答案プログラムと正解プログラムのそれぞれのトレースデータを比較する．図 3 の例の入力が“ 3，5 ”の時では，“ x=3 〃 y=5 〃 z=8 ”を比較する．EDML で設定したトレース変数に対し両者の値が一致した場合，そのトレースデータは正しいと判定する．ここで，トレースデータはプログラムの印字出力によるデータである．したがって，両者の値の完全一致による正誤判定を行うためには，答案プログラムと正解プログラムの出力形式（図 3 の正解プログラムの受講者に提示する部分）が一致している必要がある．

これらの処理を行い，CAPES は受講者の答案プログラムを正誤判定する．

3.　提案機構の概要

本章では，提案機構の構成と，プログラミング演習への適用方法について述べる．提案機構は，CAPES の答案評価機構の実行プログラム作成機能とトレースデータ作成機能を拡張し，課題に応じたトレースデータ作成を可能にするものである．提案機構の構成（図 4）は，安全な答案プログラム実行機能，変数・関数の値の調査機能，仮想機械上での答案プログラム実行機能，答案プログラムの部分実行機能の 4 つの機能からなる．提案機構は，教師が作成した EDML 文書を基に答案プログラムと正解プログラムを各機能を用いて実行し，両方の標準出力結果を従来の CAPES の機能で取り出すことで，課題に応じたトレースデータを作成可能にする．次に，4 つの機能の概要とプログラミング演習への適用方法について述べる．

安全な答案プログラム実行

CAPES では，受講者の答案プログラムを実行させて，その結果を用いて答案を評価する．この時，予期せぬプログラムの実行により，ファイルが変更・削除され，CAPES の設定が変更されてしまう可能性がある．そこで，安全な答案プログラム実行機能により，答案プログラムを資源へのアクセスが制限された環境（サンドボックス）で実行し，CAPES を保護することを可能にする．実際の演習では，教師が課題で使用する資源（例：読み込みを行うファイル“ a.txt ”）を EDML 文書に記述しておく．安全な答案プログラム実行機能は，EDML 文書の記述内容を基に，課題で使用する資源のみにアクセスを許すサンドボックスを自動で構築し，安全に答案プログラムを実行する．

変数・関数の値の調査

従来の CAPES では，変数値をプログラム実行時の印字出力を用いて評価していた．そのため，教師が指定した値を用意した関数を用いて出力するように指定しなければならない．そこで，変数・関数の値の調査機能により，デバッガを用いて答案プログラムを実行し，教師が出力文を用意せずに変数の値や関数への入力値を取得することを可能にする．実際の演習での適用方法を図 5 の加算の関数 sum を作成する問題の例を用いて説明する．まず，教師がプログラム中で使用する変数名・関数名とその内容を指定する．図 5 の例では，引数 x,y から x+y の結果を返す関数 sum を作成し，整数 x,y には標準入力の値，整数 z には x+y の結果を格納するといった名前と使用法の指定を行っている．これらの指定を用い，提案機構はデバッガの動作コマンドを生成する（図 5 のデバッガ動作コマンド）．そして，答案プログラムをデバッガ動作コマンドと入力データを用いてデバッガ上で実行し，変数の値や関数への入力値を出力結果から抜き出してトレースデータを取得する．図 5 の例の入力データ 3，5 の時では，トレースデータとして関数への入力値“ x=3, y=5 ”と各変数の最終値“ x=3，y=5，z=8 ”を取得する．関数が存在しなければ，関数への入力値“ x=3, y=5 ”の値は取得できない．このようにして取得した正解プログラムと答案プログラムのトレースデータを，トレースデータ比較機能で比較することで答案評価を行う．

仮想機械（VM）上での答案プログラム実行

プログラミング演習の中には，ネイティブコードにして動作させることができないプログラムを評価するために，VM を用いて答案プログラムの動作を確認する課題も存在する．そこで，VM 上での答案プログラム実行機能により，答案プログラムを VM 上で実行し，その実行結果を取得することを可能にする．例えばアセンブラ言語 CASL[6] の演習では，受講者はまずアセンブラプログラムを作成し，アセンブラによりオブジェクトコードを生成する．次に，COMET シミュレータとその動作コマンドを用いてオブジェクトコードをシミュレータ上で動作させることにより，実行結果を取得する．図 6 に示す加算を行う関数 sum を作成する問題の例では，問題の指定に応じた受講者の答案（CASL プログラム）を“ casl.exe ”コマンドを実行してアセンブルすることで，オブジェクトコードを作成する．次に，図 6 の例の実行コマンドでシミュレータを実行する．この時，予めファイルにしておいたシミュレータ動作コマンド列をシミュレータへの入力とする．そして，“ gr[1]=9，gr[7]=272 ”等の出力結果をトレースデータとして取得する．さらに，トレースデータ比較機能で答案プログラムと正解プログラムの出力結果を比較することで，ネイティブコードにして動作させることができないプログラムの答案評価を行う．

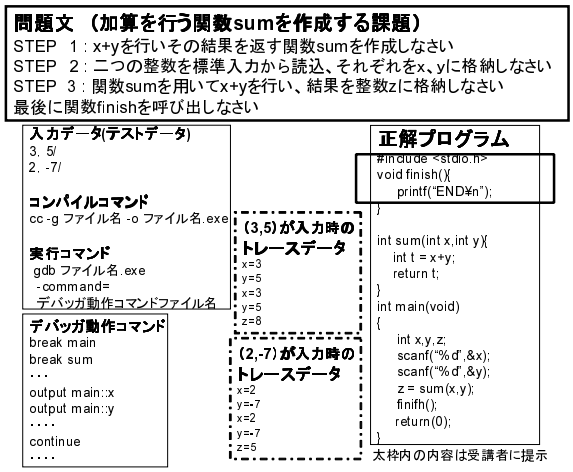

図 5 デバッガを用いた答案評価に必要なデータと出力結果の例

答案プログラムの部分実行

プログラムを部分的に作成した段階で評価可能にするために，答案プログラムの部分実行機能により，正解プログラムと受講者の答案プログラムをマージすることで，部分的に作成した答案プログラムを実行可能にする．実際の演習での適用方法を図 7 の COMET シミュレータを作成する問題の例を用いて説明する．従来の CAPES では全てのプログラムを提出しなければならなかったが，提案機構では 1 つのプログラム（例：cacos.c）だけを作成した段階で提出可能である．提案機構は，提出されたプログラム cacos.c と正解プ

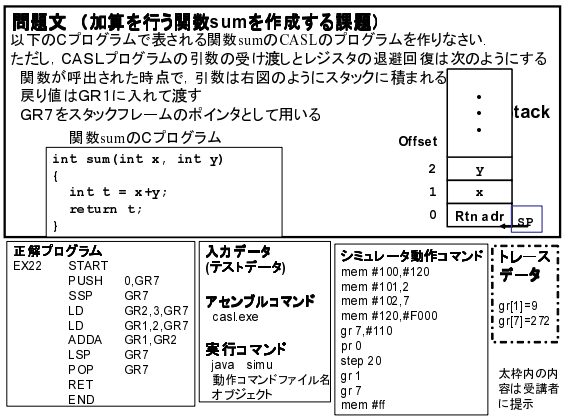

図 6　仮想機械を用いた答案評価に必要なデータと出力結果の例

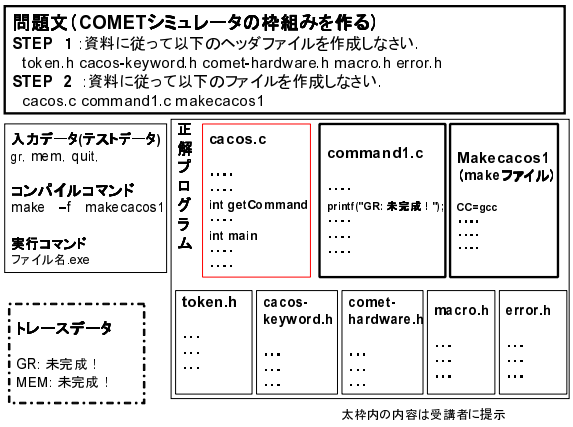

図 7　答案の部分評価に必要なデータと出力結果の例

ログラムを用いて，実行可能な答案プログラムを作成する．具体的には，提出された cacos.c と正解プログラムの cacos.c を置き換える．これより，作成した実行可能な答案プログラムをコンパイル（make[7]）・実行してトレースデータが作成でき，部分的な答案プログラムの評価が可能となる．

4. 提案機構の実現法

本章では，様々な課題に応じて答案を評価する提案機構の実現法について述べる．提案機構では，安全な答案プログラム実行機能，変数・関数の値の調査機能，仮想機械上での答案プログラム実行機能，答案プログラムの部分実行機能を課題に応じて選択し，CAPES の答案評価機構として利用することで，課題に応じた答案評価機構を実現する．提案機構を CAPES に組み込んだ答案評価機構は図 4 となる．

4.1 安全な答案プログラム実行機能

本節では，EDML で記述された内容をもとに，課

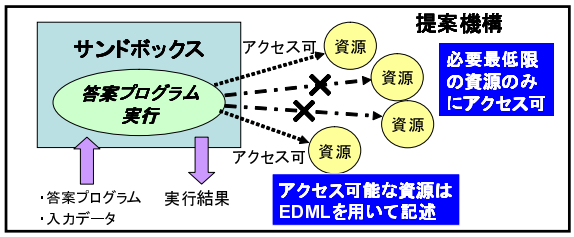

図 8　サンドボックスを備えた答案評価機構

題毎にアクセス可能な資源を求め，他へのアクセスを制限するサンドボックスを自動構築する答案評価機構の実現法について述べる．提案機構による答案評価機構は，答案プログラムをサンドボックス内で実行し，実行結果を取得して評価する（図 8）． 提案機構によるサンドボックッス構築手順は，EDML の解析と解析結果を用いたサンドボックスの構築からなる．

4.1.1 EDML の解析

EDML 文書から，答案プログラムの実行に必要な資源を求める．取得するデータは，名前（資源の名前），権限（資源にどこまでの権限を与えるかの指定），場所（資源を配置する場所），内容（資源の内容）の 4 項目とする．例として，ファイル操作の課題であれば，ファイルの名前，ファイルの実行権限，ファイルを配置するパス，ファイルの内容を取得することになる．そして，取得した内容を DB に保存する．これらの情報を，EDML を解析して自動的に取得する機能を実装することで，必要な資源を求めることが可能となる．ここで，初期状態を，プログラムの実行において，他の資源を使わない最低限の状態とする．初期状態のサンドボックスは，例えば，標準入出力を用いて四則演算を行うような初歩的なプログラム等の課題においてそのまま利用可能なものである．

4.1.2 サンドボックスの構築

サンドボックスを構築するために必要なデータを保存した後，次の手順でサンドボックスを構築する．

サンドボックスの構築 1　仮想空間の構築

コマンド chroot を用い，ファイルの名前空間を仮想化する．これにより，ファイルのアクセスを制限した仮想空間が構築できる．

サンドボックスの構築 2　資源の配置

仮想空間内に DB から取得した資源を配置する．構築機能 1-2 を実装し，これらを自動的に実行することで，毎回自動でサンドボックスを構築可能となる．サンドボックス構築後，プログラムをサンドボックス内で実行し実行結果を取得する．プログラムの実行終了後，配置した資源を削除（初期状態に戻す）するこ

とで，サンドボックスのリセットを行う．これらの手順により，サンドボックス内で答案プログラムを安全に実行可能となる．

4.2 変数・関数の値の調査機能

本節では，EDML で記述された問題の指定内容をもとに，トレースデータを作成するためのデバッガ動作コマンドを自動生成する機能の実現法について述べる．CAPES では，教師がトレースデータとして取得したい変数名・関数名を EDML を用いて記述する．提案機構では教師が指定したプログラム中で使用する変数名・関数名から，トレースデータを作成するためのデバッガを動作させるコマンドを自動生成する．これにより，教師が変数名・関数名を指定するだけで，変数の値や関数への入力値のトレースデータを取得することが可能となる．具体的には，次の手順により EDML 文書に従ってデバッガを動作させるコマンドを作成し，デバッガを動作させた結果をトレースデータとして取得する．

デバッガを用いたトレースデータ作成 1　変数名・関数名の解析

教師が EDML を用いてトレースしたい変数名を記述しておく．また，教師が問題文中にプログラム中で使用する変数名・関数名とそれぞれのプログラム中での使用法を指定しておく．また，最後にプログラムの終わりを示す関数 finish を挿入するように指定する（図 5 の受講者に提示部分）．これは，最終的な変数の値を調査するためのダミー関数である．提案機構は EDML 文書を解析することで，教師によって指定されたこれらの変数名・関数名を取得する．

デバッガを用いたトレースデータ作成 2　デバッガの動作コマンドの作成

取得した変数名・関数名を用いて，デバッガの動作コマンドを作成する．指定された関数名を用い，関数をブレイクポイントとするコマンド列を組む．さらに，各ブレイクポイントで関数への入力値を出力させるコマンド列を作成する．最後に，関数 finish の実行前に指定された変数の値を出力させるコマンド列を作成する．これにより，関数を使用しているか，関数への入力値，変数の最終値を取得するデバッガの動作コマンドが作成できる（例：図 5 のデバッガ動作コマンド）．

デバッガを用いたトレースデータ作成 3　トレースデータの取得

作成したデバッガの動作コマンドを用いて，デバッガ上でプログラムを実行する．そして標準出力結果を取得する．この時，デバッガを用いることで，プログラムへの入力とデバッガへの入力の 2 つの入力ストリームが存在することになる（図 9）．提案機構では，デバッガのオプションを利用し，デバッガへの入力は作成したファイル（デバッガ動作コマンドファイル）からの入力として区別する（図 9 の (1) のケース）．具体的には，図 5 の実行コマンド“ gdb ファイル名.exe -command=デバッガ動作コマンドファイル名 ”を用いてデバッガを動作させる．プログラムへの入力は，CAPES からの入力ストリームを用いる．

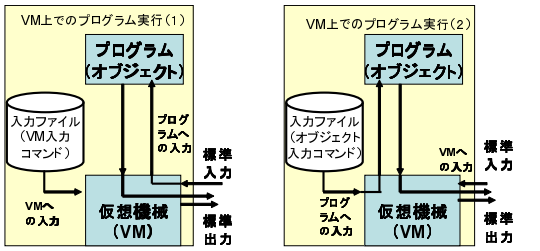

図 9　仮想機械上で答案プログラムを実行する時の入出力の関係

これらの手順により，提案機構は教師によって指定された変数・関数についてのデバッガを用いた詳細なトレースデータの作成を行うことが可能となる．

4.3 仮想機械（VM）上での答案プログラム実行機能

本節では，EDML で記述された内容（EDML 文書）をもとに，利用する VM の情報を取得し，VM 上で答案プログラムを実行可能な環境を構築する機能の実現法について述べる．VM 上で答案プログラム（オブジェクト）を実行する場合，デバッガを動作させる時と同様に，2 つの入力ストリームが存在する（図 9）．そこで，提案機構では入力データを 2 つに分割して VM に与えることで，2 つの入力ストリームを別々に扱う．また，提案機構では VM の種類に応じて，VM への入力をファイルで扱い，VM 上で動作するプログラムへの入力には標準入力を用いる（図 9 の (1) のケース）場合と，VM への入力には標準入力を用い，VM 上で動作するプログラムへの入力にはファイルを利用する（図 9 の (2) のケース）場合の 2 つの仕組みを用意している．これにより，2 つの入力ストリームをもつ VM 上で答案プログラムを動作させることが可能になる．提案機構では，教師に VM の実行コマンドと，VM への入力と VM 上で動作するプログラムへの入力の 2 つの入力定義コマンドを設定させる必要がある．次に教師によって設定された VM の情報

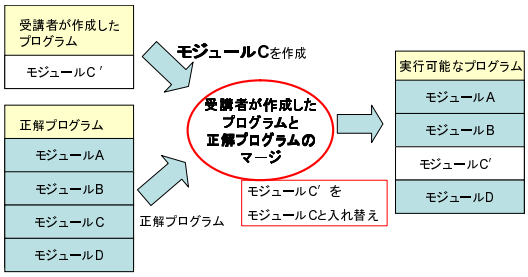

図 10 受講者の作成したプログラムと正解プログラムのマージ

から答案プログラムを VM 上で実行する手順について述べる．

VM 上での答案プログラム実行 1 VM への入力ファイルの設定

教師によって設定された VM の入力定義から，入力データを分割する．図 6 の例では，VM 上で動作するプログラムへの入力はなく，VM への入力をシミュレータ動作コマンドファイルとして渡す形であるので，シミュレータ動作コマンドの内容を“ シミュレータ動作コマンドファイル名 ”で保存する（図 9 の (1) のケース）．また，VM 上で動作するプログラムへの入力が存在する場合で，VM とオブジェクト間の入力にファイル ”a.txt ”を用いると定義してあれば，ファイル ”a.txt ”としてオブジェクトへの入力データを保存する．そして，CAPES からの入力（標準入力）として VM への入力を設定する（図 9 の (2) のケース）．

VM 上での答案プログラム実行 2 VM の実行

教師によって指定された VM の実行コマンドを用いて，答案プログラムを VM 上で実行する．図 6 の例では,“ java simu シミュレータ動作コマンドファイル名 オブジェクト名 ”として，java で作成したシミュレータ上で答案プログラムを動作させる．

これらの手順により，VM 上で答案プログラムを実行することで，ネイティブコードにして動作させることができないプログラムからトレースデータを作成することが可能となる．

4.4 答案プログラムの部分実行機能

本節では，教師が用意した正解プログラムと受講者の作成した答案プログラムをマージして，受講者の部分的なプログラムを実行する機能の実現法について述べる．本稿では特に，部分的な答案プログラムを関数単位ではなく，モジュール単位として簡単化した場合の実現法について述べる．提案機構では，CAPES が正解プログラムを用いて答案評価を行うことを利用し，受講者が作成したモジュールと正解プログラムで補完した実行に必要なその他のモジュールをマージすることで，実行可能なプログラムを作成する．これにより，複数モジュールからなる大規模なプログラムの作成課題を段階的に 1 つずつ作成する課題へと難易度を下げることができ，大規模なプログラムの作成が難しかった受講者に対し，技能習得の手助けとなる．提案機構は次の手順で正解プログラムと答案プログラムのマージを行う（図 10）．

答案プログラムと正解プログラムのマージ 1 作成されたモジュールの選択

提案機構では，教師が問題文中にプログラム中で使用するモジュール全ての名前を指定しておく．受講者は作成したモジュール（答案プログラム）とモジュールの名前を提案機構に提出する．図 7 の例では,“ cacos.c ”という名前で受講者が答案プログラムを提出する．

答案プログラムと正解プログラムのマージ 2 正解プログラムとのマージ

提案機構は，提出されたモジュール名を用いて，答案プログラムの実行に必要な残りのモジュールを抜き出し，受講者が作成したモジュール（答案プログラム）とマージする．図 7 の例では,“ cacos.c ”をみつけ，それ以外のモジュールと答案プログラムをマージする．そして，正解プログラムを利用してできた実行可能なプログラムを実行し，トレースデータを作成する．

これらの手順により，正解プログラムと答案プログラムをマージすることで，受講者が作成した 1 つのモジュール（答案）だけを評価可能となる．

5. 提案機構の評価

筆者らは 2003 年度後期から，CAPES のプロトタイプシステムを実装し，実際の演習で運用してきた．本章では，運用結果から提案機構の妥当性を検証した結果について述べる．実際に 3 年間で，名古屋工業大学での Pascal 言語のプログラミング演習（学部 1 年生），C 言語のプログラミング演習（学部 1 年生，2 部 2，3 年生），アセンブラ言語の演習（学部 2 年生）の講義で運用した．

5.1 演習の適用範囲の拡大

演習への適用結果を基に提案機構の有効性を考察する．3 年間で適用した演習の受講者数と課題数の遷移より，CAPES の適用範囲の広がりと受講者数の増加についての関係を検証した（図 11）．2003 年度は Pascal 言語の講義のみに適用し，2004 年度は C 言語

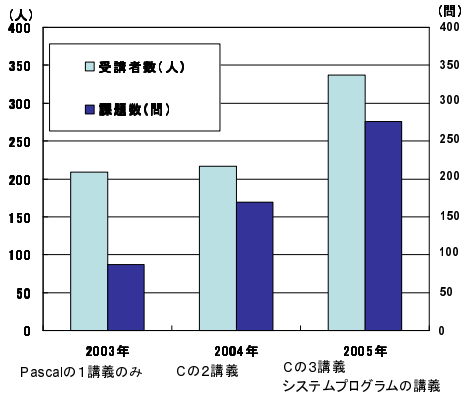

図 11 2003-2005 間における CAPES の適用状況

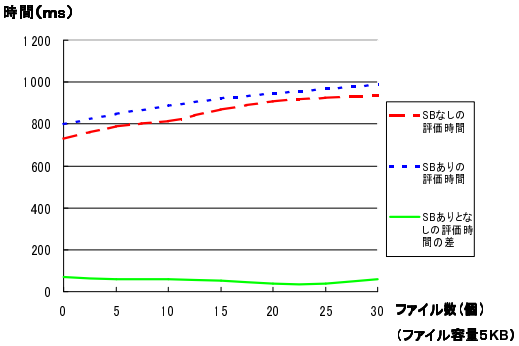

図 12 ファイル数とサンドボックス構築にかかる時間の関係

の講義にて複数モジュールからなる大規模なプログラムを作成する演習に適用範囲を広げ，2005 年度はアセンブラ言語の演習にも適用してきた．図 11 の結果から，2004 年度では C 言語の講義にも適用し講義数が増えたため，課題数が約 2 倍に増加した．受講者数は，講義毎に受講できる受講者の数が一定ではないため，あまり増加しなかった．2005 年度には，答案評価機構を改良し，システムプログラムの講義においてアセンブラ言語の演習にも適用することで，CAPES の適用範囲を広げた．また，適用した講義数が 4 つに増加し，受講者数が前年の約 1.7 倍，課題数が前年の約 1.6 倍と，両方とも大幅に増加した．これらの結果から，CAPES の答案評価機構に提案機構を導入することにより，適用範囲が広がり，多くの課題・受講者に適用できるようになったことを確認した．

5.2 サンドボックス（SB）導入における応答性

SB を用いた場合の答案評価時間と用いない場合の答案評価時間の比較を行った（図 12）．SB を用いない場合は，答案評価時に chroot による仮想環境の構築も行わないものとして時間を計測した（実験 PC のスペック：CPU クロック 3G，メモリ 1G）．

図 12 は，SB を用いた場合（提案機構）と SB を用いない場合における答案を受け取ってから評価結果を返信するまでの時間（答案評価時間）の関係を計測した図である．実験で使用した課題は，ファイルのオープンとクローズだけを行う簡単なプログラムの課題とした．図 12 から，SB を用いた場合と SB を用いない場合，つまり chroot を用いたプログラムの実行によるオーバーヘッドは 0.04s〜0.07s となり小さい値であるため，答案評価機構に提案方式を用いても性能上問題がないことを確認した．

6. おわりに

本稿では，CAPES のための答案評価機構として，安全な答案プログラム実行機能，変数・関数の値の調査機能，仮想機械上での答案プログラム実行機能，答案プログラムの部分実行機能を用い，多様な課題に対応した答案評価を可能にするシステムについて提案した．実際に提案機構の安全な答案プログラム実行機能，仮想機械上での答案プログラム実行機能，答案プログラムの部分実行機能を CAPES に導入した．そして実際の講義に適用し，提案機構の妥当性を示した．今後の課題としては，変数・関数の値を調べる機能の実装と CAPES への導入，chroot による SB が使用できない課題の評価機能の考察，課題作成時の必要条件を意識させない課題作成インタフェースの実現等があげられる．

参 考 文 献

1) 中島秀樹，高橋直久，細川宜秀，“ プログラミング学習のための QA サイクル –受講者の習得度に応じた問題自動提示メカニズム– ”，電子情報通信学会論文誌，VOL.J88-D-I，NO.2，pp439-450，2005
2) 大山恵弘，“ ネイティブコードのためのサンドボックスの技術 ”，コンピュータソフトウェア，Vol.20，No. 4，pp55-72, 2003
3) 中島秀樹，高橋直久，“ プログラミング演習支援システムのためのサンドボックスを備えた答案評価機構 ”，平成 17 年度電気関係学会東海支部連合大会（2005）
4) Richard M.Stallman, Roland H.Pesch（コスモ・プラネット訳），“ GDB デバッギング入門 ”，アスキー，東京，1992.2
5) 若林宏，“ ソフトウェアテスト手法の基本と極意 ”，株式会社　秀和システム，東京，2003
6) 小宮正好，“ CASL & COMET ”，日本理工出版会, 1988
7) Andrew Oram, Steve Talbott（菊池彰訳），“ make 改訂版 ”，オライリー・ジャパン，1997.7