競技プログラミング技能向上をめざす オンラインジャッジシステム

猪狩 知也　速水 治夫

神奈川工科大学大学院 工学研究科

s1385021@cce.kanagawa-it.ac.jp

近年，ACM 国際大学対抗プログラミングコンテストや TopCoder といったプログラミングコンテストが盛んに行われている．これらのプログラミングコンテストで出題される問題は問題文，制約，入出力の形式，入出力のサンプルから構成されている．コンテスト参加者はプログラムを実装し，プログラムを提出すると制約を満たす大量のテスト項目を試し，正誤を判定する．このような与えられた問題に対し，正確に素早く解答するプログラミングが競技プログラミングと呼ばれている．現在，プログラミングコンテストは多数存在するが学習環境が整っていなく敷居が高くなっている．競技プログラミングは普通のプログラミングとは違い，アルゴリズム自体の知識力とアルゴリズム応用力の両方が求められている．その為，知識力を強化するアルゴリズム学習とアルゴリズム応用力を鍛える問題解答学習，問題投稿による復習ができるオンラインジャッジシステムを試作した．試作システムを運用し学習した結果を報告する．

キーワード　プログラミング学習，アルゴリズム学習，オンラインジャッジ

1 はじめに

近年，ACM 国際大学対抗プログラミングコンテストや TopCoder といったプログラミングコンテストが盛んに行われている．また，プログラミングコンテストは就職活動でもプログラミング能力を調べる為に組み込まれることがあり，注目を浴びている．これらのプログラミングコンテストで出題される問題は問題文，制約，入出力の形式，入出力のサンプルから構成されている．コンテスト参加者はプログラムを実装し，プログラムを提出すると制約を満たす大量のテスト項目を試し，正誤を判定する．このような与えられた問題に対し，正確に素早く解答するプログラミングが競技プログラミングと呼ばれている．現在，プログラミングコンテストは多数存在するが学習環境が整っていなく敷居が高くなっている．競技プログラミングは普通のプログラミングとは違い，問題を解くために必要なアルゴリズム自体の知識力と問題に対してどう考え、問題の分解や要素の組み立てを行いどう表現するかといったアルゴリズム応用力の両方が求められている．知識力を強化するアルゴリズム学習とアルゴリズム応用力を鍛える問題解答学習，問題投稿による復習ができるオンラインジャッジシステムを試作した．試作システムを運用し学習した結果を報告する．

2 問題点と解決策

2.1 問題点

従来の競技プログラミング技能を向上させる手法としては，主にプログラミングコンテストに参加して問題をひたすら解く事で学習していた．この手法では問題を解く際に考える事でアルゴリズム応用力を鍛えることが出来る．しかし一方で，アルゴリズム自体の知識力向上にはあまり向いていないと考えられる．例として，素数を使用して答えを計算する問題を挙げる．この場合，素数を求めるアルゴリズムで素数を計算する箇所がアルゴリズムの知識力であり，素数をうまく使用して計算する箇所がアルゴリズム応用力の箇所になる．上記のように，アルゴリズム自体の知識を知らないと問題を解く事が出来ないか自分でアルゴリズムと同等のコードを書かなければならず，学習効果が高いといえない．上記の素数なら自力でアルゴリズムを生成出来るかもしれないが，探索やグラフアルゴリズムになってくると非常に難しい．知らないアルゴリズムの問題を解くには，先輩や友人から聞くなどの環境が整っていない場合は自ら解法のアルゴリズムについて調べ，調べたアルゴリズムを応用して解かなければならない．知っているアルゴリズムを応用するならまだしも，学んですぐに応用するのは難しいと考えられる．その為，プログラミングコンテストに参加してアルゴリズム応用力を鍛えるだけでなく，アルゴリズム自体の学習も行う必要がある．また，競技プログラミングの問題ではテスト項目にコーナーケースと呼ばれる制約ギリギリのテスト項目や見落としがちなテスト項目などの最悪なケースが含まれている．競技プログラミングの問題を解く際には，コーナーケースであっても正しい解を出力するプログラムを書かなければならない．プログラミングコンテストに参加して問題を解くだけではコーナーケースを考える力が大幅に向上するとは考えにくい．

2.2 解決策

解決策として，3 つの機能を所有するオンラインジャッジシステムを提案する．1 つ目の機能はアルゴリズム学習機能である．この機能では素数判定や幅優先探索などのアルゴリズム自体の学習を行い，アルゴリズム自体の知識力を向上させる．2 つ目の機能は問題解答機能である．この機能では他のプログラミングコンテストシステムと同じく，問題に解答してアルゴリズム応用力を向上させる．3 つ目の機能は問題投稿機能である．この機能では学習したアルゴリズムを応用して使う問題を作ることで，様々な応用を考える力がつくと考えられる．また，問題を作る際にはテスト項目を作る必要があり，コーナーケースも考えなければならないため，コーナーケースを考察する力もつくと考えられる．上記に記述した 3 つの機能を順番に学習することで競技プログラミングの能力向上に効果があると考察した．提案するオンラインジャッジシステムでの学習フローを図 1 に示す．

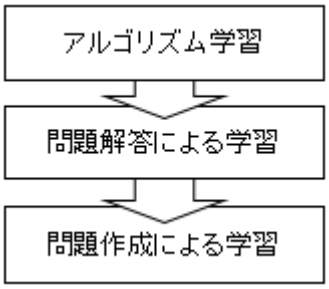

図 1 学習フロー

3 試作システム

3.1 概要

試作システムとして，アルゴリズム学習機能と問題解答学習機能，問題投稿機能を実装したオンラインジャッジシステムを作成した．試作システムは Web システムとして実装した．Web システムとして実装することにより，環境の違いによる問題を取り除くことができ，公平なジャッジを行うことができる．試作システムでは C，C++，Java，Haskell，Ruby の 5 種類のプログラミング言語を使用することが出来る．また，標準入出力でテストを行っているため言語の追加がとても容易である．

3.2 アルゴリズム学習機能

アルゴリズム学習機能では，学習したいアルゴリズムを選択して学習する機能である．アルゴリズムはカテゴリ毎に分かれており，学びたいアルゴリズムが探しやすいようになっている．図 2 に数学カテゴリのアルゴリズム一覧画面を示す．プログラミングコンテストでよく使用される素数や最大公約数を求めるアルゴリズムから，ナップザックや最長増加部分列などの動的計画法の学習などを行うことが出来る．ここでアルゴリズムを学ぶことで，次の問題解答学習で解法のアルゴリズムを知らずに解けないといった状態が起こらなくなる．

数学

#	問題名	得点	AC
41	最大公約数	100	○
42	最小公倍数	100	×
43	素因数分解	100	×
44	素数判定	100	○
45	フィボナッチ数列	100	×
70	大きな数のべき乗	100	○

図 2 アルゴリズム学習画面

3.3 問題解答機能

問題解答機能では，問題を解く事による学習が出来る機能である．前述のアルゴリズム学習機能では，どのアルゴリズムを実装すればいいかが明記されていたが，この機能では実装するアルゴリズムが明記されておらず，必要なアルゴリズムを考察して選択し，組み合わせる必要がある．上記のプロセスを踏むことで，アルゴリズム応用力の向上を狙う．図 3 に問題の例を示す．

かけざん

問題文

掛け算の答えの整数Nが1つ与えられる。

Nが答えになる a × b × cの組み合わせは何通りあるかを数えよ。

制約

-100 ≦ a,b,c ≦ 100
-1000000 ≦ N ≦ 1000000

入力

入力は1行からなり、答えの整数Nが1つ与えられる。

N

図 3 問題解答機能の問題例

上記の問題は A*B*C が N になる組み合わせが何通りあるかを求める問題である．A B C がそれぞれ-100 から 100 までと少ないので，ループ文で掛け算の結果を全て試す全探索で解を求めると O(200^3)で解く事が出来る．

3.4 自動テスト処理

ユーザがプログラムを提出すると正しい解答かどうかを判定する為にテストを行う. ユーザが提出したプログラムをサーバ上でコンパイルし, 実行ファイルを生成する. その後, 実行ファイルに対してテスト項目を標準入力で与え, 標準出力にて出力された文字列をユーザの解答として保存する. その後, 正答ファイルとユーザの解答ファイルの中身を比較して等しいかどうかを判定する. 自動テストをした際に判定結果として出力される内容を表 1 に示す.

表 1 自動テストの判定結果

判定結果	内容
Compile Error	コンパイルに失敗
Runtime Error	プログラム実行中のエラー
Not Output Error	2 秒以内に何も出力しない
Wrong Answer	正しくない結果を出力
Accepted	正答を出力

3.5 問題投稿機能

問題投稿機能では問題文, 制約, 入出力の形式, 入出力のサンプル, テスト項目, 解答をアップロードし, 問題を追加することが出来る. 学習したアルゴリズムを応用して使う問題を作ることで, 応用を考える力がつくと考えられる. また, 問題を作る際にはテスト項目を作る必要があり, コーナーケースも考えなければならないため, コーナーケースを考察する力もつくと考えられる. 問題投稿画面を図 4 に示す.

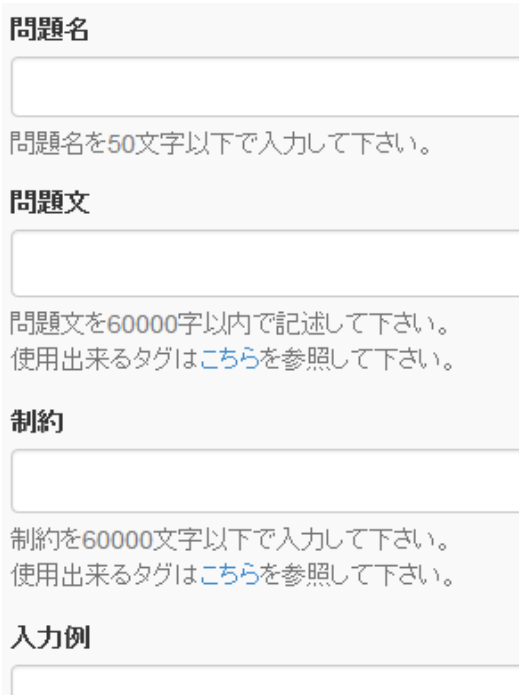

図 4 問題投稿画面

3.6 プログラミングコンテスト機能

他のプログラミングコンテストシステムと同じように, 制限時間内に何問解く事が出来るか競い合う機能を実装した.

4 評価実験

4.1 概要

評価実験は本学の 2,3,4 年生 6 人に協力して頂き評価実験を行った. 評価実験は試作システムを使用するグループと使用しないグループで 3 人ずつに分け, 使用するグループに一定の期間の間, 試作システムを使用して貰いどの程度の学習効果が発生したかを評価した. また, 使用するグループと使用しないグループ分けは ACM 国際大学対抗プログラミングコンテストの国内予選を突破したチームの 3 人は試作システムを使用しないグループに割り当て, 予選突破できなかった 3 人は試作システムを使用するグループに割り当てた. このグループ分けにより, 使用しないグループの実力に使用するグループがどの程度追い付くことが出来るかを評価することが出来ると考えられる.

4.2 試作システムでの学習

使用するグループには 2013 年 10 月から 2014 年 2 月の間使用して貰い学習を行った. アルゴリズムの学習機能と問題解答学習, 問題投稿機能は常時使用出来るようにし, 問題解答学習では主にシミュレーション, 探索, 動的計画法, ネットワークフローの問題を中心に解いて貰った.

4.3 その他の学習状況

試作システムで学習する期間中に試作システム以外でのプログラミングの学習状況を下記の表 2 に示す.

表 2 プログラミングの学習状況

番号	試作システム	その他の学習状況
1	不使用	cplusplus.com, topcoder, AtCoder, Aizu Online Judge, CodeIQ,書籍
2	不使用	node.js
3	不使用	topcoder, AizuOnlineJudge, AtCoder, 書籍
I	使用	topcoder, AtCoder, Codeforces
II	使用	topcoder, AtCoder
III	使用	なし

4.4 評価

評価は試作システム上のコンテスト形式で問題を出題し, 制限時間以内にどの程度点数を取ることが出来るかで評価した. 評価は 2 回行い, 1 回目は制限時間 75 分で行い, 問題の点数は簡単な問題から 250 点, 600 点,1050 点と設定した. 出題した問題のカテゴリは

250 点の問題がシミュレーションの問題, 600 点の問題が動的計画法の問題, 1050 点の問題が組み合わせと最小費用流の問題を出題した. 2回目は制限時間90分で行い, 問題の点数は簡単な問題から250点, 500点, 750 点, 1000 点と設定した. 出題した問題のカテゴリは250 点の問題が簡単な計算問題, 500 点の問題が動的計画法の問題, 750点の問題が幾何の問題, 1000点の問題がグラフと動的計画法の複合問題を出題した. 出題した問題は主に試作システムを使用したユーザが学習したカテゴリの問題を多く出題した. 学習したカテゴリの問題を出題することにより, 知識面でアルゴリズムが学べているかを確認するのとアルゴリズム応用力がついているかを同時に調べることが出来ると考えた.

4.5 評価結果

1回目の評価結果を表3に, 2回目の評価結果を表4に示す. 試作システムを使用したユーザを分類 A, 使用しなかったユーザを分類 B と表記している.

表 3　1回目の評価結果

番号	分類	問 1	問 2	問 3	合計
I	A	206	369	0	575
II	A	205	0	0	205
1	B	132	0	0	132
2	B	89	0	0	89
III	A	0	0	0	0
3	B	0	0	0	0

表 4　2回目の評価結果

番号	分類	問 1	問 2	問 3	問 4	合計
I	A	238	427	583	474	1722
II	A	225	368	246	0	839
1	B	231	0	160	0	391
3	B	241	0	0	0	241

表3,表4の評価結果から, 試作システムを使用したグループの方が比較的高得点である結果を得ることが出来た.

5 考察

評価実験の結果, 試作システムを使用し, アルゴリズムの学習とアルゴリズム応用力を学んだグループの方が比較的高得点である結果を得ることが出来た. 試作システムを使用していないグループは ACM 国際大学対抗プログラミングコンテストの国内予選を突破したグループで, 国内予選を突破できなかった人で構成されている試作システムを使用したグループより元々の実力は高いと考えられる. 試作システムを使用したグループの方が高得点の結果を得られたのは, 試作システムでアルゴリズムの学習とアルゴリズム応用力を効率よく学習出来たからだと考えられる. しかし一方で, 試作システムを使用したユーザで評価実験時に 1 問も解けなかったユーザが存在した. 試作システムを使用した他のユーザとの違いを比較した際に, システムの利用度に違いが見られた. 表 5 に評価実験時の得点と試作システムでの学習期間中に解いた問題数と問題を作った数を示す.

表 5 評価実験時の得点と利用記録

番号	1 回目	2 回目	正答問題数	作問数
I	575	1722	60	3
II	205	839	34	0
III	0	不参加	9	0

表5から, 試作システムをより多く利用したユーザがより高得点を取っていたことが分かった. その為, あまり試作システムを使用していなかったユーザ III は実力の向上が少なく, 評価実験時に問題が解けなかったと考えられる.

6 おわりに

競技プログラミングの能力向上には通常のプログラミングとは違い, アルゴリズム自体の知識力とアルゴリズム応用力の両方が求められていると考察してその 2 点を効率よく学ぶことにより競技プログラミングの能力が向上すると仮定し, システムを実装して評価を行った. 今回行った 2 回の評価では, 試作システムを利用することにより競技プログラミングの能力向上が見られた. しかし, 今回はほとんどの問題を筆者自身が作成したため, 使用ユーザの問題作成数が少なかった為, 問題作成による能力の向上がどの程度影響しているのかが調べられず, 今後の課題となった. また, 問題投稿機能で投稿された問題の正当性が保障出来ない課題が残った. 今後はこれらの課題を解決していき, より競技プログラミングの能力向上が出来るオンラインジャッジシステムに改善していく予定である.