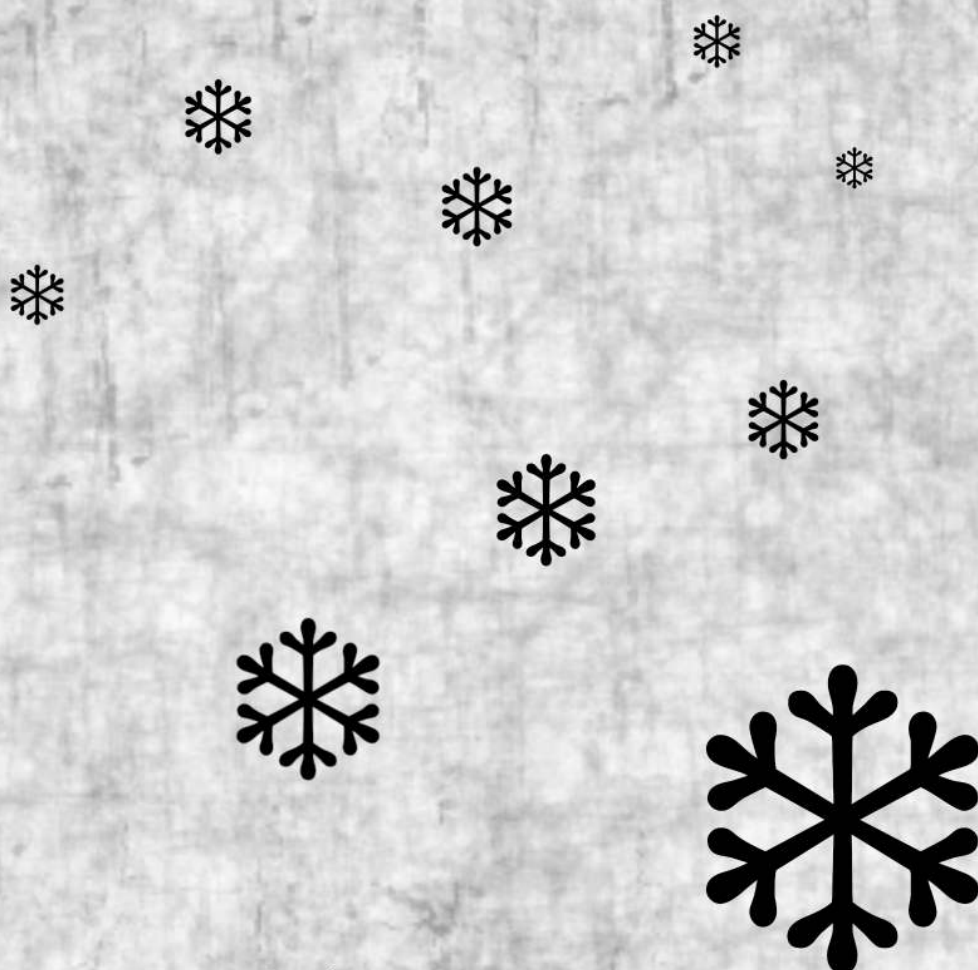


WORD

From College of Information Science



リンゴも来年には
修論を書くんだよな……号

目次

華の比文でもわかる VRChat.....	MIRA	1
自作 SAT ソルバー制作記	public_yusuke	4
Go To Eat 日記	青木 勇樹	16
映画レポート	あかぎ しげる	29
編集後記		33

華の比文でもわかる VRChat

文 編集部 MIRA

1 初めに

MIRA は VRChat^{*1}をした。必ず、美少女にならねばならぬと決意した。MIRA にはパソコンがわからぬ。MIRA は華の比文である。ラーメンを食べ、ツイッターで自分が華であると言い続けてきた。

というわけでパソコンに疎い^{*2}の比文の人間でもわかるような VRC の記事について描こうと思います。

まず VRC とはアバターを着こなして、だべったり遊んだりする SNS プラットフォームである。一応パソコンのみでも動くが、名前の通り VR 機器^{*3}を使ってやるものである。公式が用意したアバターを使って遊ぶのもできるが日本人の大半はアバターをアップロードして、それを使って遊んでいる。

2 VRChat の始め方

まず VRC の公式サイト^{*4}からアカウントを登録します。Steam アカウントのみで始めることもできますがユーザーランク^{*5}がかなり上がりにくくなります。

次に Steam のアカウントにも登録し、VRC をダウンロードします。そして起動したら先ほど登録した VRC アカウントでログインします。これで VRC に入れるようになりました。

これで美少女になれると思ったあなた残念ながらまだ早いです。

3 美少女のなり方

3.1 アバターをあげる前に

しかし始めたばかりでは自由にアバターをアップロードすることができません。VRC にはユーザーランクというシステムがあるのです。最初は Visitor で New User、User、Known User、Trusted User の順に上がっていきます。New User になるとアバターのアップロードができるようになり、User でワールドが作れるようになります。フレンドを作ったり、フレンドのところに遊びに行くのがいいでしょう。おすすめは [JP] TUTORIAL WORLD^{*6}というところに行って初心者向けの案内を受けるといいでしょう。

^{*1}VRC

^{*2}当社比

^{*3}ただ日本でメジャーで安価な PSVR とかでは動かない

^{*4}<https://hello.vrchat.com/>

^{*5}これが上がるとアバターがあげられたり、ワールド制作できる

^{*6}https://vrchat.com/home/world/wrld_bf51e60f-f372-48b1-a757-88ba8331d926

3.2 アバターのあげ方

あなたは無事 New User になり、アバターをアップロードできるようになりました。いつでも美少女になれる準備が整いました。この章ではそのアバターのあげ方を解説していこうと思います。

まず Unity の Unity 2018.4.20f1. のバージョンをダウンロードします。そして VRC 公式サイトから VRCSDK2*7を入れます。また Unity で Dynamic Bone を買いインポートすると髪が揺れたり、胸が揺れたりして幸せになります。 Booth で買ってきたアバターのファイルの中に入ってる Unitypackage のファイルを Unity の中にインポートすると新しいフォルダができるので、その中にある prefab を左上の hierarchy のところに入れると図 1 のようになります。

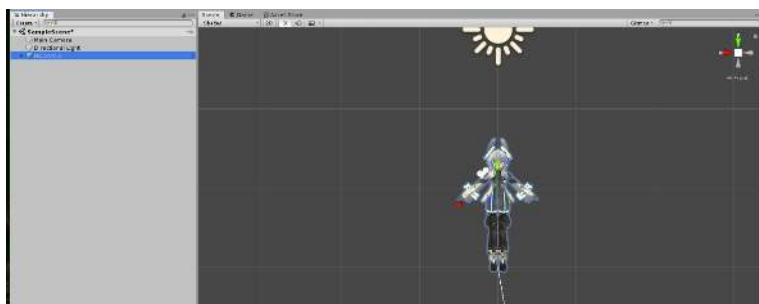


図 1 アバターをインポートしたときの様子

そしたら左上にある VRCHATSDK をクリックすると図 2 が出てくるので、build publish と書かれているところをクリックすれば完成です。

*7VRCSDK 3 が必要な場合もあります



図 2 アップロードの様子

これであなたも無事美少女になれます。というかみんなかわいくなって。No3 ちゃん*⁸はいいぞ。



図 3 No.3 ちゃん

*⁸<https://booth.pm/ja/items/1731868> で買えるぞ

自作 SAT ソルバー制作記

文 編集部 public_yusuke

1 はじめに

はじめまして、public_yusuke です！ この記事では、情報科学類の一年次で履修できる「情報科学特別演習」にて今年 5 月ごろから取り組んできた SAT ソルバーの制作について書きます。

2 SAT ソルバーの概要

2.1 充足可能性問題 (SAT) とは

SAT ソルバーとは、命題論理の充足可能性問題を解くプログラムのことです。充足可能性問題とは何でしょうか。Wikipedia における記述を引用すると [1]、

充足可能性問題（じゅうそくかのうせいもんだい、satisfiability problem, SAT）は、一つの命題論理式が与えられたとき、それに含まれる変数の値を偽 (False) あるいは真 (True) にうまく定めることによって全体の値を真にできるか、という問題をいう。SATisfiability の頭 3 文字を取ってしばしば「SAT」と呼ばれる。

とのことです。この問題は NP 完全であることでも有名です*1。命題論理式というのは、 $\wedge, \vee, \implies, \leftrightarrow, \neg$ などの記号（それぞれ論理積、論理和、論理包含、同値、否定の意）と命題変数で構成されるもので、例えば A, B, C が命題変数であるとして、「 $(A$ かつ $B)$ ならば C でない」は $(A \wedge B) \implies \neg C$ という命題論理式として表現することができます。ここでは、 A と B を True にし、 C を False にすることでこの命題論理式の値を True にすることができます。このようなとき、命題論理式 $(A \wedge B) \implies \neg C$ は充足可能 (SAT) といいます*2。

一方、命題論理式の中には、どのような変数の割り当てであっても充足できないものがあります。例えば、 $(A \wedge \neg A) \wedge B$ などは最たる例です。 $A \wedge \neg A$ の部分は、 A が True であっても False であっても必ず True にすることができないため、 B の割り当てに関わらず全体の論理式を True にすることはできません。このように充足することが不可能なとき、命題論理式は充足不可能 (UNSAT) といいます*3。

SAT ソルバーは与えられた命題論理式が SAT か UNSAT であるかを判定します。多くの SAT ソルバーでは、SAT である場合はそのときの変数の割り当てを 1 つ出力します。SAT

*1 ある問題 A があるとき、任意の NP 問題を A に多項式時間帰着可能で、かつ A が NP に含まれるとき、問題 A は NP 完全であるといえます。

*2 Satisfiable の略。他の例： $(A \vee B) \wedge (A \vee \neg B) \wedge (\neg A \vee B)$

*3 Unsatisfiable の略。他の例： $(A \vee B) \wedge (A \vee \neg B) \wedge (\neg A \vee B) \wedge (\neg A \vee \neg B)$

ソルバーの競技会である SAT Competition では、UNSAT であるときにその証明を出力することも要求されます。しかし、私のソルバーでは UNSAT のとき証明を出力しません。

2.2 SAT 問題を解く

ここからは、この問題を実際にコンピュータでどのように解くのかを取り扱います。まず最初に思いつくナイーブな方法は、与えられた命題論理式の中に登場するすべての変数の真偽値の組み合わせを全て試す方法です。

愚直に全ての変数の割り当てを試す

この方法では、変数それぞれは真と偽のどちらかの値をとるため、変数の個数を N とおくと最悪計算量は $O(2^N)$ です。もし、仮に変数が 10 個あったらどうでしょうか。このとき、検証すべき変数の割り当ての個数は $2^{10} = 1024$ ですから、(与えられた命題論理式にもよりますが) ほとんどの場合コンピュータは一瞬で SAT or UNSAT を判定できるでしょう。

では、変数が 100 個あったらどうでしょうか。仮にコンピュータが考えられる変数の割り当てのうち 10^9 通りを 1 秒で検証できるようになっていたなら、 $\frac{2^{100}}{10^9}$ 秒 ≈ 40.2 兆年ほどの時間がかかります。これでは某組み合わせのお姉さん*4のように、私たちには手に負えません。

反省

そこで、もう少し工夫して探索してみることにしましょう。冷静に考えてみると、私たちはある程度この問題を解く過程にて枝刈り*5をすることができます。しかし、その枝刈りをするためには SAT ソルバーの内部で論理式を保持するための仕組みについて、先に話す必要があります。

2.3 連言標準形

SAT ソルバーは普段利用される命題論理式そのものを持ってはいません。そのかわり、**連言標準形 (Conjunctive normal form)** を保持しています。

命題論理式では \neg という記号を使うことができます。この記号の意味は「否定」でしたね。命題変数 A があるとき、 $A, \neg A$ のことを**リテラル**といいます。

連言標準形とは、 $l_{i,j}$ がリテラルであるとき、

$$\bigwedge_i \bigvee_j l_{i,j}$$

の形式をしている論理式のことで、内側の選言（論理和のこと）の箇所を**節 (Clause)** といいます。節と節がアンドで繋がっていて、節の中はオアで繋がっている形です。

l がリテラルであるとき、単一のリテラルのみを含む節 (l) のことを**単位節 (Unit clause)** といいます。また、リテラルを全く含まない節 ($()$) のことを**空節 (Empty clause)** といいます。

*4<https://youtu.be/Q4gTV4r0zRs>

*5探索しても無駄であることがわかっている箇所については探索をしないこと。

任意の命題論理式は連言標準形に変換できることが知られている*6ので、これからは連言標準形で表されている論理式についてのみ取り扱います。なお、連言標準形は英語における各単語の先頭の文字をとって CNF と呼ばれることが多く、この記事でも以降はそうに記述します。

3 DPLL アルゴリズム

CNF を入力として受け取り、入力の SAT/UNSAT を判定する SAT ソルバーの内部で利用されるアルゴリズムには DPLL と CDCL が有名です。この章では DPLL について説明します。

3.1 DPLL の肝

DPLL は次のことを中心に成り立っています：

- **単位伝播 (Unit propagation)**：単位節 (l) が存在するとき、リテラル l を真とする変数の割り当てを自動的に行うこと
 - decision variable が n 個ある状態で単位伝播されたリテラルのレベルは n である。
 - * decision variable がない状態で単位伝播されたリテラルのレベルは 0 である。
- 変数割り当て：単位伝播がすべて完了した後、真偽値が割り当てられていない変数に対して真を割り当てる
 - このときに真偽値が割り当てられた変数のことを **decision variable** という。
- 空節 (empty clause) の存在とバックトラック：操作の結果空節が存在するとき、少なくともその時点での変数の割り当てでは与えられた論理式を充足することができない
 - このとき、以降の探索は無駄になるのでバックトラックする（後に紹介）

3.2 CNF の単純化

CNF の単純化は後に出てくる DPLL, CDCL の 2 つのアルゴリズムで共通して行う動作です。

CNF 式 F が与えられており、 F がリテラル l を持つとします。 l を True と定めたとき*7、

- l が含まれる節を削除する
- CNF 中にある $\neg l$ を削除する*8

*6Tseytin 変換を用いると入力された論理式 F の長さに対して線形の長さであるような CNF に変換することができます。この CNF は F に対して equisatisfiable、すなわち F が充足可能なとき、かつそのときに限って充足可能なものです。一般に、この CNF は F と同値ではありません。

*7変数 x について、 $l = \neg x$ なら $x = \text{False}$ 、 $l = x$ なら $x = \text{True}$ を割り当てます。

*8変数 x について、 $l = \neg x$ のとき $\neg l = x$ です。

の 2 つの操作をし、CNF 式を単純化します。これらの操作の意義を示すため、操作の例を続けます。

例として、 $F = (A \vee B \vee C) \wedge (\neg A \vee \neg C)$ とします。このとき、 A が True だと定めると、節 $(A \vee B \vee C)$ を充足することができます。このとき、この節のことは以降考える必要がないため、節を削除します。

また、右側の節 $(\neg A \vee \neg C)$ について、 $\neg A$ を削除します。なぜなら、右側の節を充足するにあたって、 $\neg A$ は False であることから、このリテラルはこれ以降この節を充足するにあたって何の役目も果たさないからです。

これらの操作を繰り返すと、最終的に「全ての節が削除されている」状態になるか、または「何のリテラルも含まない節（空節）が存在する」状態になります。実はこれらは、それぞれ SAT と UNSAT を表しています。

3.3 DPLL の基本動作

次ページに記載されている疑似コードを参考にしつつ、DPLL の全体的な流れを説明します。

まず最初に単位伝播を行います。次に未割り当ての変数を 1 つ取得し、これを x として、 x に真を割り当てて単位伝播をします。そうしてまた未割り当ての変数が出てきたら、同様のことを繰り返します。これらの過程の中で、必ず「空節が存在する」または「節の個数が 0 になる」ような状態になることが保証されています。なお、空節が存在するときのことを衝突 (**conflict**) しているといいます。前者の場合は、そのときの変数の割り当てでは UNSAT であり、後者の場合は SAT であることがわかります。SAT である場合はそこで動作を終了して良いです。

UNSAT なら、そのときの割り当てが正しくなかったことになるので、最後に割り当てた decision variable 以降の単位伝播を巻き戻し、その変数の真偽値が真だったならそれを偽にして単位伝播し、探索を継続します。もし偽だったならその一つ前に割り当てた decision variable 以降の単位伝播を巻き戻し、その変数の真偽値を見ます。このように前に戻ることをバックトラック (**backtrack**) といいます。その真偽値が真ならそれを偽にして探索を継続しますが、もし偽であったなら、さらにバックトラックします。

バックトラックの結果、最初に割り当てた変数のところまで戻ってきて、かつその真偽値が偽であるためにこれ以上バックトラックすることができない状態になったとき、与えられた CNF は充足不可能であるということが決定します。

与えられた CNF が充足可能なら、そのようなことが発生する前に SAT となるような変数の割り当てが見つかります。

DPLL の疑似コード

実際に DPLL の疑似コード [2] を示したほうがわかりやすいので、掲載します。

なお、 $F|_l$ は、リテラル l を True としたときに F を単純化した結果の CNF 式を示します。

また、 ρ は、真であると割り当てられたリテラルの集合にあたり、 $\rho \cup \{l\}$ はリテラル l

Algorithm 1 DPLL(F, ρ)

入力：CNF 式 F と空の割り当て ρ

出力：UNSAT、または F を充足する割り当て

Begin

$(F, \rho) \leftarrow \text{UnitPropagate}(F, \rho)$

if F が空節を含む **then**

return UNSAT

end if

if F に節がない **then**

ρ を出力

return SAT

end if

$l \leftarrow \rho$ で割り当てられていないリテラル

if DPLL($F|_l, \rho \cup \{l\}$) = SAT **then**

return SAT

end if

return DPLL($F|_{\neg l}, \rho \cup \{\neg l\}$)

End

sub UnitPropagate(F, ρ)

Begin

while F が空節を含まないが単位節 (l) を持つとき **do**

$F \leftarrow F|_l$

$\rho \leftarrow \rho \cup \{l\}$

end while

return (F, ρ)

End

に真を割り当てするという意味です。

3.4 パフォーマンス

この疑似コードを参考にしてゴリゴリ実装します*⁹。

例えば CNFgen*¹⁰ で \$ `cnfgen gop --complete 8` として生成できる変数 56 個、節 372 個のケースは手元のコンピュータで 1 分弱かかって UNSAT と判定することができました。

$\frac{2^{56}}{10^9}$ 秒 ≈ 115.7 日 ですから、ナイーブな解法では約 115.7 日かかってやっと UNSAT で

*⁹<https://github.com/private-yusuke/sat-d/blob/master/source/solvers/dpll.d>

*¹⁰<https://massimolauria.net/cnfgen>

あることがわかります。それと比較すれば、1 分弱でその判定ができるのはすごいです。

4 CDCL アルゴリズム

4.1 CDCL とは

DPLL でもかなりの高速化ができましたが、例えば数独を SAT 問題に変換 [3] して解こうとしたとき、数百個もの変数が現れます。その他にも、実用的な問題を解こうとしたときにはより多くの変数や節が登場することは不可避です。これは、DPLL をもってしても太刀打ちできません。

そのため、CDCL アルゴリズムを用いてさらなる高速化をします。CDCL は Conflict-driven clause learning の略で、現代的な SAT ソルバーの高速化において非常に大きな貢献をしたアルゴリズムです。基本的には DPLL をベースとして機能するので、単位伝播や CNF 式の単純化については DPLL と同等のを行います。

CDCL の主な目的は「衝突の原因」にあたるものを新たな節として学習することで探索の枝刈りをすることです。この節を**学習節 (learned clause)**といいます。学習節は、探索が継続する中でいつでも有効に働くポテンシャルを秘めているのが特徴的です^{*11}。DPLL では衝突が起きたらその都度バックトラックしていましたが、これでは局所的にしか枝刈りが効かず、後の探索中にまた同じような状況になってバックトラックが発生する可能性があります。一方、CDCL の学習節は衝突したそのときに限らず、後の探索中に同様の理由によって起こりうるような衝突の発生を防ぐことができます。

CDCL における衝突 (conflict) は DPLL と同じもので、CDCL では単位伝播などによる変数割り当ての結果、リテラル $l, \neg l$ がどちらも真であるように直前に割り当てられた結果起こります [2]。

4.2 CDCL の基本動作

CDCL の疑似コード

次ページに記載されている疑似コード [2] を参考に CDCL の全体的な流れを説明します。

まず、DecideNextBranch では、まだ真偽値が割り当てられていない変数に真を割り当てています。具体的には、すべての変数に 1 から順に正の自然数を対応させたとき、未割り当ての変数のうち、対応する正の自然数が一番小さいものに真を割り当てることにします^{*12}。その内側の while ループの最初では Deduce 関数を呼び出していますが、これは今までの単位伝播に対応するものです。

Deduce 後、ソルバーが衝突状態になっている^{*13}ならば、その衝突の原因を分析して学習節を追加します (AnalyzeConflict)。ただし、ここで単位伝播のみで矛盾が生じたとわ

^{*11}ブラウザにおけるキャッシュのような気持ちです。

^{*12}変数選択のヒューリスティクスもいくつか存在します。例えば、より多くの節に現れているリテラルを優先的に割り当てるといったものがあります。これらも高速化の鍵を握っています。

^{*13}空節ができたということ

Algorithm 2 CDCL

入力：CNF 式 F 出力：UNSAT、または F を充足する割り当て**Begin**

```
while TRUE do
  DecideNextBranch
  while TRUE do
    status ← Deduce
    switch status do
      case CONFLICT
        blevel ← AnalyzeConflict
        if blevel = -1 then
          return UNSAT
        end if
        Backtrack(blevel)
      case SAT
        現在の割り当てを出力
        return SAT
      case OK
        break
    end while
  end while
```

End

かった（blevel = -1）なら追加は発生しておらず、UNSAT と判定して終了します。そうでないときは、Backtrack 関数を呼び出し、矛盾する前まで（具体的には blevel と同じレベルを持つような decision variable に真偽値が割り当てられた直後に）バックトラックします。これによって衝突の直接の原因となった割り当ての 1 つ前の選択をやりなおすことができます。なお、blevel は必ずしも矛盾の 2 つ前のレベルになるとは限らないので、それと比べればさらに前へバックトラックする可能性があります、そのときはより良い本質的な選択がそこにあったということになります。

Deduce 関数が SAT を返すのは、与えられた CNF に現れる全ての変数に対して衝突が発生せずに真偽値の割り当てが完了したときになります^{*14}。

それ以外の状況になったとき（OK）は単位節が存在しなくなった場合に当たるので、内側のループを抜け、DecideNextBranch が呼ばれます。

^{*14}全ての節が削除された状態に対応します。

4.3 衝突までの道筋

CDCL ソルバーの動作は **Implication graph** を考えることで理解しやすくなります*¹⁵。これは、decision variable や、単位伝播によるリテラルの導出のされ方を保持する有向非巡回グラフです。それぞれの頂点には真が割り当てられているリテラルが一对一に対応しています。節 $(l_1 \vee l_2 \vee \dots \vee l_n \vee l)$ について、 l_1, l_2, \dots, l_n がすべて偽であるとき単位伝播で l が真となるので、これをグラフ上で表すため $\neg l_1, \neg l_2, \dots, \neg l_n$ から l に対して辺が伸びています。

最終的にすべての変数がグラフの頂点として現れ、かつ衝突が起きていないとき、与えられた CNF は SAT であると決定できます。

CDCL の動作例を見ながら implication graph がどのように作られていくのか見てみましょう。与えられた CNF 式 F は

$$F = (2 \vee 3 \vee 5) \wedge (-1 \vee -3 \vee 4) \wedge (-3 \vee -4 \vee 5) \wedge (-1 \vee 3 \vee 5) \wedge (1 \vee -2 \vee -3) \wedge (-2 \vee 3 \vee 4) \wedge (-2 \vee -4 \vee -5)$$

であるとしします。この式には 5 つの変数が登場しますが、それらに 1 から 5 の正の整数 x を対応させています。なお、 $x, \neg x$ はそれぞれリテラル $x, \neg x$ を意味します*¹⁶。

まず最初に **DecideNextBranch** が走ります。ここでは、未割り当ての変数のうち、対応する正の自然数が一番小さいものに真を割り当てることにしているので、リテラル 1 が真となります。すると implication graph にリテラル 1 の頂点が追加されます。ここで CNF 式の単純化を行うと

$$F = (2 \vee 3 \vee 5) \wedge (-3 \vee 4) \wedge (-3 \vee -4 \vee 5) \wedge (3 \vee 5) \wedge (-2 \vee 3 \vee 4) \wedge (-2 \vee -4 \vee -5)$$

となります。最初の F は全ての節の長さが 3 であったのに対して、リテラル -1 が削除されて長さが 2 になった節があることに注目してください。また、リテラル 1 が含まれていた節が削除されていることに注目してください。

この状態で **Deduce** 関数が呼ばれますが、単位節がないので何も起きません。また、衝突していないし、全ての変数に対して真偽値の割り当てが終了したわけでもないので、ループが一周して **DecideNextBranch** が呼ばれ、同様に リテラル 2 に真を割り当てます。すると、

$$F = (-3 \vee 4) \wedge (-3 \vee -4 \vee 5) \wedge (3 \vee 5) \wedge (3 \vee 4) \wedge (-4 \vee -5)$$

となり、まだ単位節がないので、リテラル 3 にも真を割り当てて

$$F = (4) \wedge (-4 \vee 5) \wedge (-4 \vee -5)$$

となります。ここで初めて単位節 (4) が現れたので、**Deduce** 関数による単位伝播が発生し、

$$F = (5) \wedge (-5)$$

となります。**Deduce** 関数による単位伝播は、ここではより右側にある節が優先されるように実装してあるので、リテラル -5 が単位伝播されます。すると、 $F = ()$ となります。

*¹⁵MiniSat 含め数多くの高速なソルバーはこれを内部に持っていませんが、私のソルバーは保持しており、ここにある説明にあることを忠実に実行しています。

*¹⁶このように、SAT 問題では一对一に変数と正の整数を対応させ、それぞれの否定についてはマイナス記号を付けて表現する方法が一般的です。

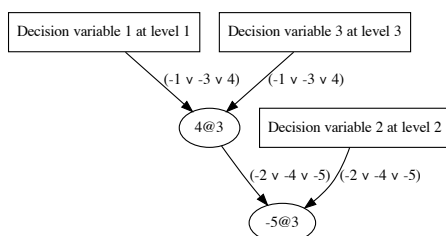


図 1 Implication graph の例

この時点での implication graph は図 1 のようになります。3 つの decision variable がありますが、これらに対応する頂点は入次数が 0 であり、ここからグラフが延びていることがわかります。 $n@m$ の形で表されている頂点は、単位伝播によって導出されたリテラル n と、そのレベル m を表しています。これらの頂点は入次数が 1 以上であり、そこに入ってくる辺のラベルには、単位伝播で利用された、単純化が適用されていないオリジナルの節が対応しています。4@3 の親は、4@3 を単位伝播で導く原因となったリテラルに対応しています。

例えば、Decision variable 1 at level 1 と Decision variable 3 at level 3 からの辺が刺さっている 4@3 の頂点は、2 つのリテラル 1, 3 が真であることから、節 $(-1 \vee -3 \vee 4)$ に対し単純化が適用された結果 (4) になって単位伝播されたことによりリテラル 4 が導出されたことを表しています。

今、 $F = ()$ ですから、空節が存在しています。このようなとき、最後に真となったリテラル -5 の否定 5 が削除されたことで空節が発生しているので、5 も単位伝播のときと同じようにグラフに追加し、辺を生やします*¹⁷。すると、図 2 のように implication graph 上に 5, -5 のどちらも存在することになり、Deduce 関数内では衝突したという判定が下されると同時に、衝突の直接の原因となっている 5, -5 から衝突を表す特別な頂点 Λ に辺が延ばされます。このように、衝突している状態を表しているグラフのことを **Conflict graph** といい*¹⁸ます。衝突が発生したので、Deduce 関数は CONFLICT を返します。

衝突の分析とバックトラック

AnalyzeConflict について詳しく解説します。この関数は、衝突の原因を抽出し、それを学習節としてソルバーに追加する関数です。conflict graph は、衝突が含まれる conflict side と、その原因となったものが含まれる reason side の 2 つにカットすることができます。reason side 側にはすべての decision variable を含む必要があります。このカットの仕方に

*¹⁷実装上ではリテラル l を割り当てたときに単位節 $(-l)$ の中の $-l$ は消さずに保持しておくようになっています。その結果、以降の単位伝播のプロセスで $-l$ も対象となり、 $-l$ に真が割り当てられる直前に、グラフ上にすでに l が乗っていることを判定することではじめて衝突したとみなしています。

*¹⁸conflict graph には implication graph よりも厳しいルール (Λ を除く全ての頂点は Λ への道が存在する、など) がありますが、ここでは長くなるので省略します。詳しくは [2] を参照してください。

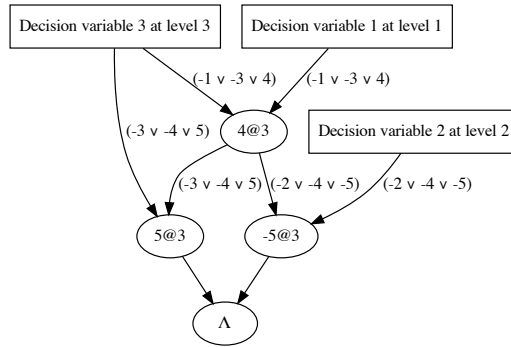


図 2 Conflict graph の例

よって、学習節が確定します。

学習節を得るためには、まず reason side に属する頂点 r から conflict side に属する頂点 c へと延びる辺 $r \rightarrow c$ の r にあたるものを全て集めます。次に、それらの頂点に対応するリテラルの否定を集め、「または」の記号 (\vee) で繋ぎます。これで得られたものが学習節になります。

今回はグラフを reason side と conflict side にカットする仕方に **1UIP cut** を採用します*¹⁹。これによると、得られる学習節は $(\neg 1 \vee \neg 2 \vee \neg 3)$ になります。これをソルバーに追加することで、以降の探索で与えられた CNF 式 F に加えて考慮される対象となり、同様の原因によって生じる衝突を回避することができます*²⁰。

AnalyzeConflict で返される **blevel** は、学習節に含まれるリテラルのレベルのうち 2 番目に高いレベルになります*²¹*²²。これを引数として **Backtrack** 関数が呼び出されます。この関数は、**blevel** のレベルを持つ decision variable へ真偽値が割り当てられた直後まで、CNF 式や学習節の単純化、変数の割り当て、implication graph の頂点や辺の状態を巻き戻す機能を持っています。

その結果、真であるリテラルは 1, 2 のみとなり、CNF 式 F は

$$F = (2 \vee 3 \vee 5) \wedge (-3 \vee 4) \wedge (-3 \vee -4 \vee 5) \wedge (3 \vee 5) \wedge (-2 \vee 3 \vee 4) \wedge (-2 \vee -4 \vee -5)$$

の状態に戻ります。ここで、先程学習した節は CNF の単純化によって (-3) になるので、これが単位伝播されます (図 3)。

これ以降の implication graph の推移は、図 4 のようになります。途中で再度衝突が発生

*¹⁹様々なカットの仕方がありますが、1UIP cut は効果的で、実装もしやすいです。詳しくは [2] を参照してください。

*²⁰これが成立するのは resolution という導出原理との関係があつてのことです。

*²¹学習節が単位節になるときもあります。このときは、レベルとして 0 が採用されます。

*²²このレベルにバックトラックすることで、学習節に含まれる、ある 1 つのリテラルにのみ真偽値が割り当てられていない状況になります。そのため、結果的に単位伝播で前回とは反対の真偽値が割り当てられることが予想できます。

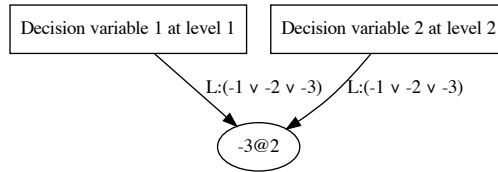


図 3 学習節による単位伝播 (L: となっているのに注目)

し、学習節が追加されていることがわかります。

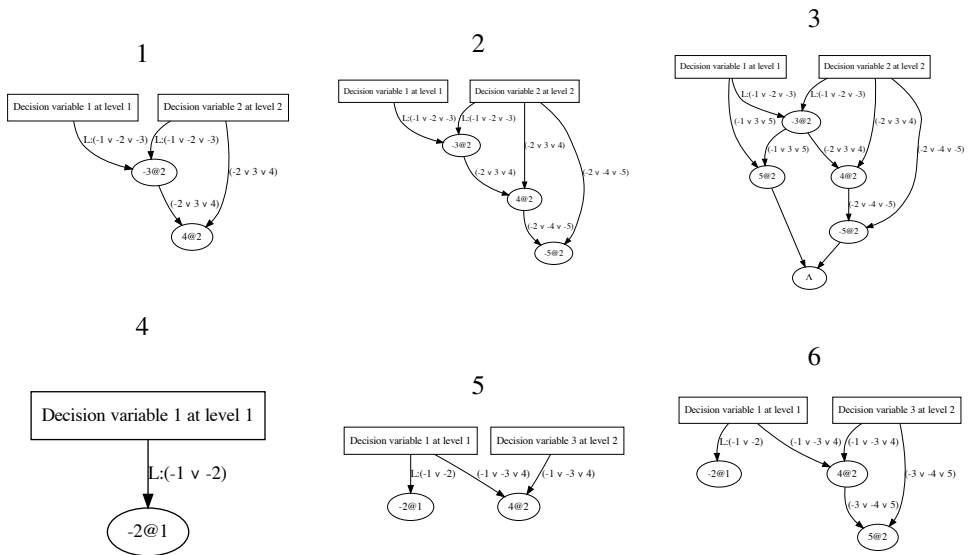


図 4 implication graph の推移

図 4 の 6 番目のグラフが最後のグラフで、このように全ての変数が頂点に現れ、かつ衝突がない状態になっているため、**Deduce** 関数は **SAT** を返します。これで、与えられた CNF は SAT であるということがわかりました。

4.4 パフォーマンス

CDCL の実装は DPLL と比較して非常にバグらせやすく、「SAT・UNSAT の判定は正しいのに DPLL より遅い」という謎の期間が一時期あって結構辛かったのですが、1UIP cut の実装を見直したところ、テストケースによっては数百倍も高速になったり、そもそも DPLL では解けなかった問題も数秒で答えが出るようになりました。例えば、DPLL のパフォーマンスの箇所で生成したケースは手元のコンピュータで約 15 ミリ秒、`$ cnfgen gop --complete 19` として生成できる変数 342 個、節 6004 個のケースは約 1.3 秒かかって UNSAT と判定することができました。

5 おわりに

SAT ソルバーに使われているアルゴリズムのうち有名なものには DPLL と CDCL があり、今回私が実装したソルバーではどちらも利用できるようになっています。また、Tseytin 変換も実装してあるので、任意の命題論理式の充足可能性を判定できるようになっています。

命題論理まわりのことについては大学以前から興味があったので、今回このように数学的な対象とコンピュータの結びつきを自らの手で体験することができて楽しかったです。また、私は小さなプログラムはよく書くものの、全体のことを考えつつ細部を実装していく経験がまだまだ足りないなということを実装していく上で体感させられました。途中でバグらせて3週間ぐらい進捗が滞ったり、特に CDCL をうまく実装したつもりが実質的には DPLL と同等のことをしていたということもあったりして悩んだ時期もありましたが、最終的にソルバーの動作を理解するのに必要な概念や考え方を把握しつつ1から実装できて嬉しかったです。

他にも変数選択のヒューリスティクスや実装上の工夫^{*23}によってまだまだ遥かに高速化できるのですが、ひとまずここまで進めることができました。

6 参考文献

- [1] Wikipedia. 充足可能性問題 — Wikipedia, The Free Encyclopedia. <http://ja.wikipedia.org/w/index.php?title=%E5%85%85%E8%B6%B3%E5%8F%AF%E8%83%BD%E6%80%A7%E5%95%8F%E9%A1%8C&oldid=62281013>. [Online; accessed 17-November-2020]. 2020.
- [2] Frank van Harmelen et al. *Handbook of Knowledge Representation*. San Diego, CA, USA: Elsevier Science, 2007. ISBN: 0444522115.
- [3] Inês Lynce and Joël Ouaknine. “Sudoku as a SAT Problem”. In: *PROCEEDINGS OF THE 9 TH INTERNATIONAL SYMPOSIUM ON ARTIFICIAL INTELLIGENCE AND MATHEMATICS, AIMATH 2006, FORT LAUDERDALE*. Springer, 2006.

7 ソースコード

<https://github.com/private-yusuke/sat-d/> を参照してください。D 言語で書かれています。

^{*23}watched literal と呼ばれる実装上の工夫や、変数割り当てによるソルバーの状態の変更を、差分を持っておくことで直接キャンセルできるようにするなど

Go To Eat 日記

文 編集部 青木 勇樹

本記事執筆中にオンライン予約でポイントを付与する事業のポイント付与額が予算上限の 616 億円に達してしまい、オンライン予約でのポイント付与が早くも終了してしまいました。

また、新型コロナウイルス感染症の再拡大に伴い、プレミアム付き食事券の発行が一時停止しています。

記録のために利用方法の記述を残しておきますが、予約をしてもポイントは付与されず、食事券の購入もできません。なお、すでに付与されたポイントと食事券の利用はできます。オンライン予約に関する記述は参考程度にお読みください。

みなさん、Go To Eat キャンペーンを活用していますか？

Go To Eat キャンペーンとは、新型コロナウイルス感染症の感染拡大に伴う外出の自粛等の影響により、甚大な影響を受けている飲食業に対し、期間を限定して需要喚起を図る官民一体型のキャンペーンです*1。

販売額にさらに 25% も上乗せされるプレミアム付き商品券、オンライン予約サイトで対象飲食店を予約すると一人あたり 1000 円分ものポイントをもらえる、といった前代未聞のお得なキャンペーンです。

そんな Go To Eat キャンペーンの活用方法と、私が利用した Go To Eat 対象店を日記形式でご紹介します。なお、価格は税込表示と税抜表示が混在しています。ご了承下さい。カッコ内の表示がないものは、ポイントの日本円相当金額、または支払金額です。

1 プレミアム付き食事券

Go To Eat キャンペーンの一つ、プレミアム付き食事券についてご紹介します。

各都道府県のキャンペーン運営事務局が発行するプレミアム付き食事券を買うと、販売額の 25% が上乗せされた額が上乗せされます。例えば、12500 円分の商品券を 10000 円で購入できます。お得すぎますね！ 本記事では、茨城県版プレミアム付き食事券について紹介します。他の都道府県については各都道府県のキャンペーン事務局のウェブサイト等をご覧ください。

プレミアム付き食事券が利用できる店舗は茨城県のキャンペーン事務局のホームページで検索できます。

*1 よくあるご質問 | 農林水産省 GoToEat キャンペーン <https://gotoeat.maff.go.jp/faq/>

【公式】Go To Eat キャンペーンいばらき | 茨城県 <https://www.gotoeat-ibaraki.com/>

1.1 買い方

茨城県版プレミアム付き食事券は、10000 円 (12500 円分) 単位で購入できます。

1. セブン-イレブンに行きます。
2. マルチコピー機を探します。
3. 「チケット」を押します。
4. プレミアム付き食事券の大きなバナーを押します。
5. 枚数を選択し、「OK」ボタンを押します。
6. 氏名、電話番号を入力します。
7. 印刷された受付用紙をレジに持っていきます。
8. 代金を現金またはクレジットカードで支払います。

1.2 使い方

会計時に食事券を提示します。

会計前に食事券を利用できるか確認すると良いでしょう。その際に食事券をお預かりしたいとお願いされる場合があります。

なお、食事券利用時にお釣りは出ません。

1.3 販売・利用期限

令和 3 年 1 月 31 日（日）23:29 まで販売しています。なお、販売数量上限に達した場合は、予告なく販売終了となります。

食事券の利用期限は 令和 3 年 3 月 31 日（水）です。

2 オンライン予約(終了しました)

本事業のポイント付与額が予算上限の 616 億円に達してしまい、オンライン予約でのポイント付与が終了してしまいました。ポイントの付与については参考程度にお読みください。なお、ポイントの利用は可能です。

二つ目の Go To Eat キャンペーンの施策であるオンライン予約についてご紹介します。

オンライン飲食予約サイト経由でキャンペーン期間中に予約し、来店すると、次回以降に予約サイト経由でポイントがもらえます。そのポイントが、一人あたり最大 1000 円！お得すぎますね！

ランチタイムの予約で、一人あたり 500 円、15 時以降の予約で一人あたり 1000 円相当のポイントが付与されます。

一回の予約あたり最大 10 人まで、つまり最大 10000 円まで利用できます。

キャンペーンを使える予約サイトは、ぐるなび、食べログ、Yahoo! ロコなど、13 社 15 サ

イトです。一覧はキャンペーンページをご覧ください。

農林水産省 GoToEat キャンペーン <https://gotoeat.maff.go.jp/>

2.1 使い方

Yahoo!ロコを例に解説します。

1. お店のページを開きます。
2. 「Go To Eat ポイントもらえる・使える」の表示があることを確認します。



3. コースを選択します。



4. 予約画面で、日付、人数、時間を設定します。



5. ポイントを利用する場合はポイント設定を行います。
6. 予約した時間通りにお店に向かいます。
7. 「ネットで予約した〇〇 (予約者の苗字) です」と言います。
8. 一人当たり 1000 円以上注文します。
9. 通常通り支払います。もしポイントを利用する場合はポイントを利用する旨を伝え、確認メールなどポイント利用額がわかるものを提示してください。
10. 翌日以降にポイントが付与されます。付与されるタイミングはサイトごとに異なります。

2.2 付与・利用期限

ポイントの付与は最長 2021 年 1 月末まで行われます。

ポイントの利用期限は最長で 2021 年 3 月末までですが、ポイントサイトによって異なります。また、ポイントの形態 (ポイントを消費できるサービス、ポイントの名称) もポイントサイトによって異なります*2。ご利用の予約サイトや、付与されたポイントの表示をよくご確認ください。

3 日記

価格の記述は諸事情によりレシートが残っていないため、記憶または領収証を頼りにして書いています。間違っていたらごめんなさい。

頼んだメニューの記載は私の注文分のみです。

3.1 10/7 牛角 テクノパーク桜店

- 頼んだメニュー: 早割 1980 円 (税抜)
- 人数: 7 人
- 予約サイト: Yahoo! ロコ

初の Go To Eat です。18 時までに入店すると、なんと 1980 円 (税抜) で 90 分食べ放題の「早割」キャンペーンを利用しました。安い分、ビビンバ、牛タンなど、人気メニューは食べ放題には含まれていませんが、腹を肉で満たすにはピッタリ！ Go To Eat ポイントを 1000 ポイントもらえるので、実質 980 円でお腹いっぱいになります！

3.2 10/8 割烹一の矢

- 頼んだメニュー: いくら丼 1400 円 (税抜)
- 人数: 3 人
- 予約サイト: ホットペッパー グルメ

一の矢宿舎を北に越えたところにある割烹料理店です。定食から鰻重まで、幅広い和食料理を提供しています。その中で特にコスパが良いメニューが、日替わり A 定食です。ボリュームも味もバッチリの定食がなんと 950 円 (税抜) で頂けます。それでも Go To Eat 対象です。ポイントを利用すると、たった 45 円で食事ができます。

そんなに安いと逆に申し訳なくなると思いますし、せっかくの Go To Eat ですから高いものを食べて幸せになりましょう。

割烹と聞いて価格が高そうだと感じてしまう、また、少し大学から遠いので、Go To Eat キャンペーンが始まるまで行ったことのない方も多いのではないのでしょうか。これを機に是非足を運んでみてください。

*2例: ホットペッパーグルメはポイント付与日の翌々月末まで飲食店でのみ消費できる、Yahoo! ロコはポイント付与日から 60 日間まで飲食店でのみ消費できる

3.3 10/11 割烹一の矢

- 頼んだメニュー: 日替わり B 定食 1100 円 (税抜)
- 人数: 3 人
- 予約サイト: ホットペッパー グルメ

また来ました。こんなにコスパがいいと来るしかありません。店員さんに顔と名前を覚えられました (店舗には氏名と電話番号が通知されます)。

今日の日替わり定食はピーマンの肉巻き、鮪の刺身、ベーコンとジャガイモの煮物でした。ピーマンの肉巻きは小さい頃から好きなんですよ。懐かしい味でした。

3.4 10/12 安安 五反田西口店

- 頼んだメニュー: 安安コース 2000 円程度
- 人数: 2 人
- 予約サイト: ホットペッパー グルメ

名前の通り安かったです。牛角の早割と同じく安い肉なのだろうと思っていたのですが、予想以上に肉が食べ応えがあり、いい意味で裏切られました。

3.5 10/13 びすとり椿々 追越店

- 頼んだメニュー: サングリア、ジントニックなど
- 人数: 2 人
- 予約サイト: ホットペッパー グルメ

安価なダイニングバーです。安価な上に Go To Eat を使ったので、たくさん飲んでしまいました。

3.6 10/14 煉瓦亭

- 頼んだメニュー: 90 分食べ放題 3850 円 (税抜)
- 人数: 3 人
- 予約サイト: ホットペッパー グルメ

私の業務で重宝させていただいているコードフォーマットツール Prettier のコミットに奢りました。OSS に還元しないとね！

高めの焼肉店です。Go To Eat がなければ行けませんでした。高いだけあって、肉が厚切りで、味も深みとコクがありました。学生にとってはなかなか食べられない貴重な味です。

とても美味しかったので、調子に乗って牛タンを注文し過ぎてしまいました。追加料金を払って持ち帰ることになってしまいました。お腹とよく相談して注文しましょう。

なお、ネット予約の場合やクーポンを併用した場合、クレジットカードを利用できません。ご注意ください。

3.7 10/16 割烹一の矢

- 頼んだメニュー: うに丼 うどんセット 1500 円 (税抜)
- 人数: 2 人
- 予約サイト: ホットペッパー グルメ

3 回目の来店です。

味噌汁やお吸い物がつく丼物は、100 円追加で味噌汁をうどんに変更できます。

3.8 10/22 牛角 テクノパーク桜店

- 頼んだメニュー: 早割 1980 円 (税抜)
- 人数: 8 人
- 予約サイト: Yahoo!ロコ

2 回目の来店です。早割です。後輩をたくさん呼んで奢ると、Go To Eat ポイントと T ポイントとクレジットカードのポイントが大量にたまります。

3.9 10/28 一太郎

- 頼んだメニュー: 食べ放題 1900 円 (税抜)
- 人数: 3 人
- 予約サイト: ホットペッパー グルメ

もんじゃとお好み焼きのお店です。お好み焼きは基本的には関西の料理で、関東に来てから食べておらず、久しぶりに食べることができて良かったです。

食べ放題メニューは、男性が 1900 円、女性が 1700 円と記載されていますが、一人当たりではなく、テーブルの男女比で決定されるそうです。なお、男女の判定基準と、男女が半々になった時の対応についてお伺いしたのですが、回答はありませんでした。プログラミングで言う所の、例外処理漏れです。

3.10 10/30 牛角 テクノパーク桜店

- 頼んだメニュー: 早割 1980 円 (税抜)
- 人数: 3 人
- 予約サイト: Yahoo!ロコ

3 回目の来店です。早割です。メニューが少ないのでそろそろ飽きました。

本日から茨城県版プレミアム付き食事券の先行予約者向けの配布が始まりました。この券が使えるかどうか尋ねたところ、(当然ながら) 食事券利用者第一号で、券をお預かりしたいと言われました。新型コロナウイルスも、Go To Eat キャンペーンも、お店にとって突然のことでしょう。協力を求められたら応じましょう。

3.11 11/3 エル・トリート つくば学園店

- 頼んだメニュー: チキン&ビーフファヒータ、メキシカンタコライス、モスコミュールなど
- 人数: 3 人
- 予約サイト: ホットペッパー グルメ

メキシコ料理のファミレスです。複数人向けで、一品あたりの価格は高めの品が多いです。なので、あるカップルと一緒に行きました。

ファヒータがとても美味しかったです。トルティーヤとビーフの相乗効果で、とても美味しく、メキシカンなスパイスが効いていて、メキシコの雰囲気味わえました。

プレミアム付き食事券を利用しました。

3.12 11/9 割烹一の矢

- 頼んだメニュー: カキフライ炊き込みご飯定食 1500 円 (税抜)
- 人数: 4 人
- 予約サイト: ホットペッパー グルメ

4 度目の来店です。とてもカリッと揚げられたカキフライは食べ応えがありました。角煮がとても柔らかかったです。

3.13 11/10 牛角 テクノパーク桜店

- 頼んだメニュー: 早割 1980 円 (税抜)
- 人数: 3 人
- 予約サイト: Yahoo!ロコ

もう 4 度目の来店です。早割です。飽きました。

Go To Eat ポイントを 5000 ポイント、牛角でのみ使える「期間固定 T ポイント」を 334 ポイント、残りの 1200 円を PayPay で決済しました。「超 PayPay 祭り『牛角』で超おトクキャンペーン」で 120 ポイント、期間固定 T ポイントが 300 ポイント、そして Go To Eat ポイントを 3000 ポイント獲得しました。利用ポイントから獲得ポイントを引くと、3114 円、一人あたり 1038 円になりました。

3.14 11/10 居酒屋 LATIGO

- 頼んだメニュー: 日本酒 1 合 500 円 (税抜)、ハイボール 450 円 (税抜)
- 人数: 3 人
- 予約サイト: ぐるなび (同伴の友人が予約しました)

ノリで二次会をしてしまいました……

麺 the Tokyo に隣接している居酒屋です。お酒とまぜそばを一緒に楽しめます。お腹いっぱいだったのでまぜそばは食べませんでした。

3人で合計 4010 円 (税込) で、4000 円分のプレミアム付き食事券で支払いました。プレミアム付き食事券の 25% 分を考慮すると、実質 3210 円で、さらに一人当たり 1000 ポイントも付与されるので、実質 210 円、一人当たり 70 円でした。お店側に申し訳なくりますね。

3.15 11/10 びすところ椿々 追越店

- 頼んだメニュー: 白ワイン 399 円 (税抜)、海鮮カルパッチョ 399 円 (税抜) など
- 人数: 3 人
- 予約サイト: ホットペッパー グルメ

さ……三次会です……

3人で合計 3620 円 (税込) でした。3000 円分のプレミアム付き食事券で支払いました。プレミアム付き食事券の 25% 分を考慮すると、実質 3020 円で、さらに一人当たり 1000 ポイントも付与されるので、実質 20 円、一人あたり 6.6 円でした。あの……

4 よくある質問

Q. 使えるお店は？

A. 食事券が使えるお店はキャンペーン事務局のホームページで検索できます。オンライン予約でポイントが付与されるお店は、予約サイトの表示やキャンペーンサイトをご覧下さい。時期や条件によっては利用できない場合がございます。

Q. 食事券とオンライン予約のポイントは併用できますか？

A. できます。オンライン予約ではポイントをもらいつつ 1000 円未満の飲食しかししない、いわゆる「錬金術」が規制されましたが、食事券の併用による実質的な錬金術は問題ありません。

Q. オンライン予約しなくても食事券を利用できますか？

A. できます。

5 写真

Go To Eat キャンペーンを使った時の写真です。どうぞ。



図1 びすとり椿々 追越店 ワイン



図2 割烹一の矢 カキフライ炊き込みご飯定食



図3 エル・トリート つくば学園店 チキン&ビーフファヒータ



図4 エル・トリート つくば学園店 メキシカンタコライス



図5 一太郎



図6 安安 五反田西口店



図7 割烹一の矢いくら丼

映画レポート

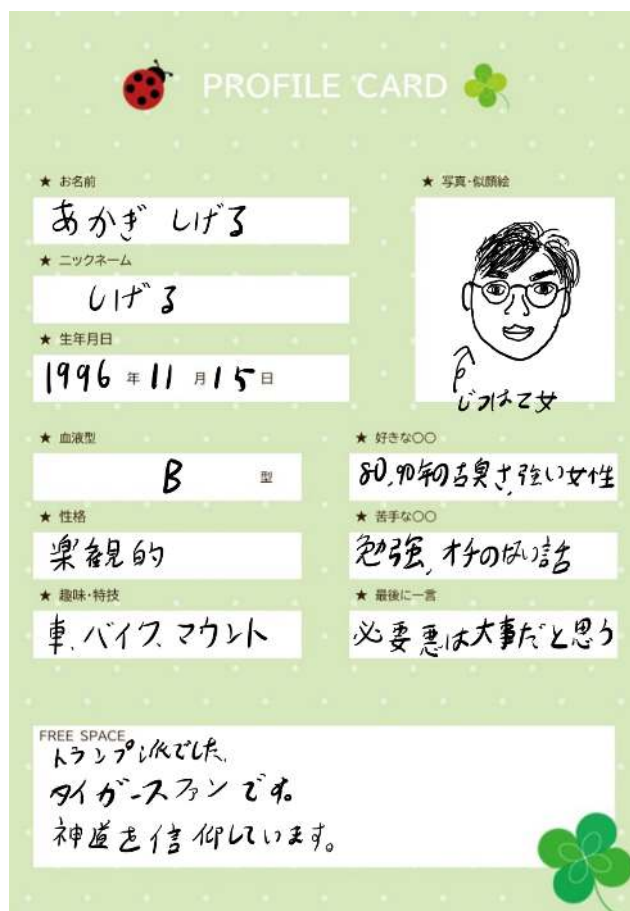
文 編集部 あかぎ しげる

1 はじめに

皆さんこんにちは。あかぎしげるです。こんongo時世映画でも見て心を磨いてみてはいかがでしょうか？

2 自己紹介

映画を紹介する前に僕がどんな人間であるかを紹介しなければ、僕がどのような気持ちでこの映画を見てどう思ったのかは伝わりづらいだろう。そんなわけで自己紹介カードを書いてみることにする。



PROFILE CARD

★ お名前
あかぎ しげる

★ ニックネーム
しげる

★ 生年月日
1996 年 11 月 15 日

★ 血液型
B 型

★ 性格
楽観的

★ 趣味・特技
車, バイク, マウレト

★ 写真・似顔絵
じろふて女

★ 好きな○○
80,90年代の古臭い強い女性

★ 苦手な○○
勉強, オチの偽り話

★ 最後に一言
必要悪は大事だと思う

FREE SPACE
トランポリンで決。
タイガースタンです。
神道を信仰しています。

3 映画たち

今回は読者の気分によってどんな映画をどんな順番で見ればいいのか紹介していこうと思う。

3.1 落ち込んでいるとき～

研究や人間関係、大学生は様々なことに悩まされて生きている。そんな落ち込んでいるときは以下の2本を見れば悩みごとは吹き飛ぶだろう。

デスプルーフ in グラインドハウス

クエンティン・タランティーノ監督の作品である。70,80年代のB級映画をオマージュして作られ、監督の特徴である意味のない会話が長々続いている。そのような中にも映画としての盛り上がりや期待感をもたせてくれる。

突然始まるカーアクションも秀逸で、手負いの心に純粋な映画の楽しさを教えてくれるだろう。

いつだってやめられる7人の危ない教授たち

学生の悩みの種の大きな種として担当教員つまり教授が挙げられるだろう。この映画はそんな教授を題材とした映画である。大学をクビになるなど、不遇な状況になった教授たちが合法ドラッグを製造・販売することになる。教授らの専門分野を活かしドラッグは大人気になっていくというストーリーである。

いつもは愛のムチで泣かせてくる教授たちも裏で泣いているのだろう、自分だけが落ち込んでいるわけではないと思える。

3.2 恋破れたとき～

大学生が悩んでいる理由はだいたいこれであろう。そんなあるあるネタには以下の3本で過去の異性を忘れて頂こう。

シェイプ・オブ・ウォーター

発話障害を抱えている女性が職場の研究所で半魚人のような新しい生物と出会う。手話を交えて生物との関係が徐々に親密になっていくというストーリーである。

新しい出会い、これまでは考えられない新しい価値観、そんな事を教えてくれる。また一般的な世界観ではなく自己投影が難しいため、自分の心がより傷つく事も少ないだろう。

愛・アムール

ミヒャエル・ハネケのパルムドール受賞作品である。老夫婦のジョルジュとアンヌが集合住宅の一室で暮らしていたが、アンヌが発作を起こし二人の生活が崩れていく。映画の中で老い、愛、思いやりがリアルに描かれている。

映画として純粋に面白いし、過去の異性について悩み倦めるのではなく、より大きな悩みを考えることでちっさな悩みは忘れてしまおう。

プラダを着た悪魔

主人公はジャーナリスト志望でファッション誌業界に就職したアンドレアとファッション誌の鬼編集長ミランダである。アンドレアはファッションに興味はなかったが、ミランダのアシスタントとして懸命に働く中でファッションと仕事の面白さに目覚めていくという話である。

愛・アムールで悩みのすり替えを行った。あとはこの悩みを忘れるため、仕事に打ち込むビジネスマンになれば綺麗サッパリである。また、アン・ハサウェイのきれいなファッションに触発される事で自分磨きもできるだろう。

3.3 自分探しをしたいとき～

大学生が自分探しをするときたいい場合はインドに行くらしいが、残念ながら現在インドへの渡航は制限されている^{*1}。お家留学したい、そんな人に向けた製作国がインドのいわゆるボリウッド映画3本を紹介する。

^{*1}https://www.anzen.mofa.go.jp/info/pcinfectionspothazardinfo_001.html#ad-image-0

ロボット

科学者のバシーガラン博士は自分に似せた人形ロボットを完成させる。しかしそのロボットに感情を与えなければデリカシーに欠け、感情を与えると完璧な仕事をこなせないといった不完全なものだった。ゴミとして捨てられてしまったロボットがその後暴走してしまうというストーリーである。

題目の通りインド産のロボット技術やCG技術について笑いながら学べる。歌、踊り、笑いなどボリウッドが詰まった映画で、お家留学の初日にインドを感じるには一番の映画だろう。

ムトゥ踊るマハラジャ

ラージャー家に仕えるムトゥは歌って踊れる馬使い手して重宝されていた。しかし主人が一目惚れした女優とムトゥがいい感じになってしまうというストーリーになっている。

面白く踊りやダンスがいい娯楽映画であるが、単なる娯楽ではなく人々の幸福や慈愛とはなんだろうと考えさせられる。留学ついでに自分探しに行った大学生と同じ気分を与えてくれるだろう。

マダム・イン・ニューヨーク

主人公の主婦は家族の中で唯一英語ができないことを悩んでいる。そんな中親戚の結婚式の手伝いで単身渡米することになり、スキマ時間に家族に秘密で英会話に通い始めるというあらすじである。

古典的な価値観の中で育ってきた主婦が新しい環境で成長していくことで、周りの環境がいかに大事かを考えさせられる内容となっている。お家留学終わりに日本に留まり続けてはダメだと思えるだろう。

4 おわりに

みなさん、いかがでしたか？ いやぁ、映画って本当にいいもんですね。それでは、またご一緒に楽しみましょう。

情報科学類誌



From College of Information Science

リンゴも来年は修論書くん
だよな……号

発行者 情報科学類長

編集長 広瀬智之

筑波大学情報学群

情報科学類WORD編集部

制作・編集 (第三エリアC棟212号室)

2020年12月22日 初版第1刷発行

(256部)