

2025	生徒商研	41006	鳥栖商	江戸川区総合文化センター	11/11 - 11/12
------	------	-------	-----	--------------	---------------

令和7年度第33回

全国高等学校生徒商業研究発表大会

鳥栖商 数理最適化 プロジェクト

TOSUSHO Mathematical Optimization Project



佐賀県 | 佐賀県立鳥栖商業高等学校 | 情報処理部

生徒氏名

大江 陽菜乃 | 福島 蘭 | ティグリージェシカ | 岡 和希 | 黒田 琉斗

指導者氏名

今村 豊記

目次

1 ▶	はじめに	2
2 ▶	数理最適化と 0-1 整数計画問題	2
3 ▶	研究の仮説	3
4 ▶	研究内容	3
Lv.1	ナップサック問題	4
【例題 1】	4
Lv.5	割り当て問題	6
【例題 2】	6
【例題 3】	9
Lv.Max	巡回セールスマン問題	10
(1)	巡回セールスマン問題とは	10
(2)	巡回セールスマン問題のモデル化	10
(3)	巡回セールスマン問題の Web アプリ	12
(4)	人間の経験 vs Web アプリ	16
(5)	原因の分析と改善	17
(6)	人件費・燃料費削減のシミュレーション	18
5 ▶	プロジェクトの検証・評価	19
(1)	仮説の検証	19
(2)	流通業者（ヤマト運輸）からの視点より	19
(3)	I T システム開発会社（プロシップ）からの視点より	19
(4)	流通業者以外（J T B・J R 九州・自治体等）の視点より	20
(5)	成果・課題・改善と展望	21
6 ▶	おわりに	22
7 ▶	参考文献等	23

1▶ はじめに

私たち鳥栖商業高校情報処理部はこれまでプロジェクションマッピング、VR、アバターなど、広報・広告の一環となるようなビジュアル要素の高いコンテンツを生み出してきました。今回は、これまでのような派手さはありませんが、商業科の基本に立ち返り、企業経営に資する研究活動を行いました。

私たちの日常や経営活動において、意思決定を迫られる場面が多くあります。例えば、限られたお小遣いでどの洋服を買うか、どの進路を選べば自分の夢に近づけるか、などです。今回の研究をはじめのきっかけは「長期間欠席した友人が課題の提出をする際、校内各所にいらっしゃる先生方に、どの順番、どの経路でまわった方がよいかを悩んでいた」というエピソードからです。

この問題をコンピュータやアプリを使って解決できないか検討したところ「**数理最適化**」という分野があることを知りました。この分野を学習することは、日常生活だけでなく、商業高校生にとって必要な**経営活動における意思決定**につながるのではないかと考えました。

2▶ 数理最適化と 0-1 整数計画問題

「**数理最適化**」の目的は**制約条件下で最適な解**を算出することです。

数理最適化の例として、商業科の科目「ソフトウェア活用」で学習する「**線形計画法**」という単元内の「**生産計画の最適化**」があります。複数の原材料を用いて複数の製品を製造する場合、原材料の利用上限数を制約条件とし、利益を最大化するための最適な製造数を求めるという問題です。

2つ目の例として、複数の配送先を最も効率よくまわるための経路や輸送車を割り当てる「**配送計画の最適化**」があります。訪問可能時間や輸送車の最大積載量が制約条件となり、合計配送時間・距離を最小化することが目的となります。

他にも数理最適化で求められる問題は多数あります。授業では「線形計画法」を学びましたが、私たちは今回「**0-1 整数計画問題**」というものに挑戦しました。

「線形計画法」の最適解は実数で答えることができるのに対し、「整数計画問題」では解を**整数に限定**します。例えば、解の対象が人数の場合、「5.8 人」などの解が算出されても、「0.8 人」という人数は現実的には割り当てることができないため、5 人か 6 人に丸めるなどして対処する必要があります。ただ、丸めた解が最適解であるかどうかは不明なため、**解を整数に限定して最適解を求めるのが「整数計画法」**です。

さらに「0-1 整数計画問題」では解を **0 と 1 に限定**します。**0 はその選択をしない、1 はその選択をする**という意味です。解が 0 と 1 だけに限定されるので簡単に思われがちですが、実数の解は公式等で比較的簡単に答えられるのに対し、0 と 1 の解に限定した場合、**解の組合せ**を考えるので、**計算量が膨大になる**場合があります。下表にその計算量の一例を示します。

入力サイズ	計算時間 (100MIPS の計算機による実行時間)
10	2.1×10^{-5} 秒
20	1.05×10^{-2} 秒
30	10 秒
40	3.05 時間
50	130 日
100	26798 宙齡 (1 宙齡はビッグバンから現在までの時間)

組合せの解は
計算量が
爆上がり!!



指数時間アルゴリズムの計算時間 『組合せ最適化とアルゴリズム』 P22 より

「数理最適化」という分野を知り、商業を学ぶ私たちが意思決定を科学的に行えるということは、とても有益なことであると考えました。最初のきっかけは「先生方をまわる最適経路」でしたが、「0-1 整数計画問題」にも種類が多くあります。私たちの研究は、教科書の範囲を超えたこの「0-1 整数計画問題」を学習し、コンピュータでモデル化して、人間が考えた解より優れた解を求めることができるかが主な内容となります。

3 ▶ 研究の仮説

日常生活や経営活動の意思決定の場面を 0-1 整数計画問題としてモデル化し、人間が考えた解より優れた解をコンピュータで求めることができる。

4 ▶ 研究内容

既に 0-1 整数計画問題の分野は確立しており、様々な問題が提示され、先行研究も数多くあります。私たちの研究の主な内容は最後に示す「巡回セールスマン問題」ですが、その前に日常生活や経営活動に置き換えやすい 2 つの問題と、私たちが取組んだ解決方法を紹介します。

Lv.1 ナップサック問題

「ナップサック問題」は、0-1 整数計画問題の中でもシンプルな問題です。限られた容量を持つナップサックにどれだけ好きなおやつを入れることができるかを考えます。

n 個のおやつを大きさ $a_i (i = 1, \dots, n)$ 、おやつ i の満足度を $c_i (i = 1, \dots, n)$ とします。おやつ i の大きさの合計がナップサックの容量 b を超えないという制約条件のもと、おやつ i の満足度を最大にする組合せを考えます。

変数 x_i の値は 0 と 1 であり、おやつ i を選択するとき 1、おやつ i を選択しないとき 0 になります。このように 0 と 1 のみに限定する変数を「0-1 変数」と言います。この問題の例題を紹介します。

【例題 1】

例題

1

A 君は 10 の容量を持つナップサックを持っています。

おやつは 5 種類ありますが、全ては入りきれません。ナップサックの容量を超えないように 5 種類から選ぶ必要があります。

下表のようなおやつ i の大きさ a_i と満足度 c_i のとき、最も満足度が大きくなる組合せを求めます。なお、おやつは 1 つずつです。



	グミ 	アメ 	クッキー 	せんべい 	ラムネ 
大きさ	4	2	5	3	1
満足度	3	1	4	5	2
選択	x_1	x_2	x_3	x_4	x_5

x_1 から x_5 は、0 と 1 のいずれかを選択し、0 のときはそのおやつをナップサックに入れない、1 のときは入れるという意味です。

【例題 1 の定式化】

最大化 $z = 3x_1 + x_2 + 4x_3 + 5x_4 + 2x_5$ — 満足度計 (x_i が 1 のとき満足度を加算)

制約条件 $4x_1 + 2x_2 + 5x_3 + 3x_4 + x_5 \leq 10$

— 大きさ計 (x_i が 1 のとき大きさを加算) は容量以下

x_i は 0-1 変数 ($i = 1, \dots, 5$)

【Microsoft Excel によるモデル化とソルバーの利用】

Excel を使って問題をモデル化しました。問題の各係数を Excel に入力し、「大きさ計」と「満足度計」は、SUMPRODUCT 関数で求めます。「選択」に「1」が入力された分、各おやつ i の大きさ a_i と満足度 c_i の合計が求められるようにしています。なお、SUMPRODUCT

関数とは、各範囲の対応する数値の積を合計した結果を返す関数です。授業や検定試験では扱われませんが、かなり便利な関数です。

	A	B	C	D	E	F	G	H	I
1		グミ	アメ	クッキー	せんべい	ラムネ			
2	大きさ	4	2	5	3	1			
3	満足度	3	1	4	5	2			
4	選択						←持っているものは「1」		
5									
6	ナップサック	≥		大きさ計					
7	10			0					
8				=SUMPRODUCT(B2:F2,B4:F4)					
9				満足度計					
10				0					
				=SUMPRODUCT(B3:F3,B4:F4)					

バイナリ (0 か 1) に制限

目的は満足度計の最大化

目的セルの設定:(I) \$C\$10
目標値: ☒ 最大値(M) ☐ 最小値(N) ☐ 指定値:(V) 0

変数セルの変更:(B) \$B\$4:\$F\$4

制約条件の対象:(U)
 \$B\$4:\$F\$4 = バイナリ
 \$C\$7 ≤ \$A\$7

大きさ計はナップサックの容量以下に制限

解決方法: 滑らかな非線形を示すソルバー問題には GRG 非線形エンジン、線形を示すソルバー問題には LP シンプレックス エンジン、滑らかではない非線形を示すソルバー問題にはエボリューションエンジンを選択してください。

ヘルプ(H) 解決(S) 閉じる(Q)

次にこのモデルを Excel の追加機能であるソルバーを用いて最適解を求めました。なお、ソルバーは教科書にも掲載されており、先述した線形計画法の問題で使い方は学習済です。

目的は満足度の合計値であるセル C10 の最大化です。

変数セルはセル B4～F4 です。0-1 変数なので制約条件として「バイナリ」を選択します。

大きさ計はナップサックの容量以下に制限します。

ソルバー実行！

	A	B	C	D	E	F	G	H	I
1		グミ	アメ	クッキー	せんべい	ラムネ			
2	大きさ	4	2	5	3	1			
3	満足度	3	1	4	5	2			
4	選択	0	0	1	1	1			←持っているものは「1」
5									
6	ナップサック	≥	大きさ計						
7	10		9						
8			=SUMPRODUCT(B2:F2,B4:F4)						
9			満足度計						
10			11						
11			=SUMPRODUCT(B3:F3,B4:F4)						
12									

ソルバーを実行した結果、「クッキー」「せんべい」「ラムネ」を選ぶと**大きさ計 9**で**ナップサックの容量 10 以下**を満たし、**最大の満足度計 11**を得ることができます。

例題ではわかりやすく「ナップサック」の例をそのまま用いていますが、ナップサックにおやつを入れるのにここまで考えることは現実的にはありません。この問題は現実的な**経営活動として置き換えることができます**。例えばおやつをビジネス上の**プロジェクト**や**金融商品**と考え、各プロジェクト・金融商品には費用 $a_i (i = 1, \dots, n)$ と効果 $c_i (i = 1, \dots, n)$ があります。**予算 b の範囲で最大の効果を得るためにはどれを選択すべきか**といった問題に適用できます。

Lv.5 割り当て問題

次に「割り当て問題」を紹介します。これは**1 人の従業員に 1 つの業務を割り当てるときに効果が最適となるような組合せ**を考える問題です。

n 人の従業員を n 種類の業務に割り当てる場合を考えます。従業員が各業務を完了できる時間はそれぞれ異なり、従業員 $X_i (i = 1, \dots, n)$ が業務 $Y_j (j = 1, \dots, n)$ を完了する時間を a_{ij} と置きます。各業務を複数人で行うことはなく、必ず 1 人に 1 つの業務を割り当てるとき、時間の合計が最も小さくなるような組合せを求めます。0-1 変数 x_{ij} は従業員 X_i を業務 Y_j に割り当てるとき 1、割り当てないとき 0 になります。

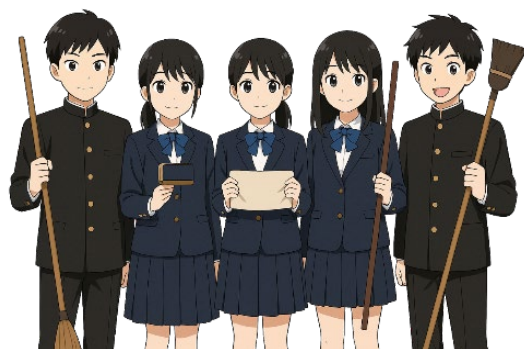
【例題 2】

例題 2

学校の掃除分担において、5 人の生徒に 5 種類の掃除を割り当てる場合を考えます。

下表のように各生徒の掃除の完了時間(分)を推測したとき最も時間の合計が小さくなる組合せを求めます。

なお、1 人の生徒には必ず 1 つの掃除を割り当てます。



	床を掃く 	床拭き 	机運び 	窓拭き 	黒板消し 
1 号	3	13	5	8	5
2 号	10	16	3	4	6
3 号	8	11	7	10	8
4 号	5	8	15	4	3
5 号	13	10	11	3	4

各生徒に割り当てる掃除を選ぶために 0-1 変数 x_{11} から x_{55} を下表のように準備し、これらは次の意味を持ちます。

$$x_{ij} = \begin{cases} 1 & (i\text{番目の人に}j\text{番目の掃除を割り当てる}) \\ 0 & (i\text{番目の人に}j\text{番目の掃除を割り当てない}) \end{cases} \quad (i = 1, \dots, 5; j = 1, \dots, 5)$$

	床を掃く 	床拭き 	机運び 	窓拭き 	黒板消し 
1号	x_{11}	x_{12}	x_{13}	x_{14}	x_{15}
2号	x_{21}	x_{22}	x_{23}	x_{24}	x_{25}
3号	x_{31}	x_{32}	x_{33}	x_{34}	x_{35}
4号	x_{41}	x_{42}	x_{43}	x_{44}	x_{45}
5号	x_{51}	x_{52}	x_{53}	x_{54}	x_{55}

【例題2の定式化】

最小化 $z = 3x_{11} + 13x_{12} + 5x_{13} + 8x_{14} + 5x_{15}$
 $+10x_{21} + 16x_{22} + 3x_{23} + 4x_{24} + 6x_{25}$
 $+8x_{31} + 11x_{32} + 7x_{33} + 10x_{34} + 8x_{35}$
 $+5x_{41} + 8x_{42} + 15x_{43} + 4x_{44} + 3x_{45}$
 $+13x_{51} + 10x_{52} + 11x_{53} + 3x_{54} + 4x_{55}$ 一割り当てた時間の合計

制約条件 $x_{11} + x_{12} + x_{13} + x_{14} + x_{15} = 1$ $x_{21} + x_{22} + x_{23} + x_{24} + x_{25} = 1$
 $x_{31} + x_{32} + x_{33} + x_{34} + x_{35} = 1$ $x_{41} + x_{42} + x_{43} + x_{44} + x_{45} = 1$
 $x_{51} + x_{52} + x_{53} + x_{54} + x_{55} = 1$

一ある生徒が行う掃除はどれか1つなので各行の合計は1

$x_{11} + x_{21} + x_{31} + x_{41} + x_{51} = 1$ $x_{12} + x_{22} + x_{32} + x_{42} + x_{52} = 1$
 $x_{13} + x_{23} + x_{33} + x_{43} + x_{53} = 1$ $x_{14} + x_{24} + x_{34} + x_{44} + x_{54} = 1$
 $x_{15} + x_{25} + x_{35} + x_{45} + x_{55} = 1$

一ある掃除を行う生徒は誰か1人なので各列の合計は1

$x_{ij} = 0$ または $x_{ij} = 1$ ($i = 1, \dots, 5; j = 1, \dots, 5$)

この問題も Excel でモデル化し、ソルバーを用いて最適解を求めました。

【Microsoft Excel によるモデル化とソルバーの利用】

	A	B	C	D	E	F	G	H	I	J	K	L
1		床を掃く	床拭き	机運び	窓拭き	黒板消し						
2	1号	3	13	5	8	5						
3	2号	10	16	3	4	6						
4	3号	8	11	7	10	8						
5	4号	5	8	15	4	3						
6	5号	13	10	11	3	4						
7												
8		床を掃く	床拭き	机運び	窓拭き	黒板消し	計	制約条件				
9	1号						0	1				
10	2号						0	1				
11	3号						0	1				
12	4号						0	1				
13	5号						0	1				
14	計	0	0	0	0	0						
15	制約条件	1	1	1	1	1						

目的は
掃除時間計
の最小化

時間計
0

=SUMPRODUCT(B2:F6,B9:F13)

目的セルの設定:(I) \$J\$9

目標値: ☐ 最大値(M) ☒ 最小値(N) ☐ 指定値:(L)

変数セルの変更:(B) \$B\$9:\$F\$13

制約条件の対象:(U) \$B\$9:\$F\$13 = バイナリ

\$B\$14:\$F\$14 = \$B\$15:\$F\$15
\$G\$9:\$G\$13 = \$H\$9:\$H\$13

バイナリ(0 か 1)
に制限

各人の計が 1、
各掃除の計が 1
になれば 1 人 1 役

ソルバー実行！

	A	B	C	D	E	F	G	H	I	J	K	L
1		床を掃く	床拭き	机運び	窓拭き	黒板消し						
2	1号	3	13	5	8	5						
3	2号	10	16	3	4	6						
4	3号	8	11	7	10	8						
5	4号	5	8	15	4	3						
6	5号	13	10	11	3	4						
7												
8		床を掃く	床拭き	机運び	窓拭き	黒板消し	計	制約条件				
9	1号	1	0	0	0	0	1	1				
10	2号	0	0	1	0	0	1	1				
11	3号	0	1	0	0	0	1	1				
12	4号	0	0	0	0	1	1	1				
13	5号	0	0	0	1	0	1	1				
14	計	1	1	1	1	1						
15	制約条件	1	1	1	1	1						

時間が最小となる
割り当てが決定！

時間計
23

=SUMPRODUCT(B2:F6,B9:F13)

掃除の割り当てを例としましたが、実際には高校の掃除にかかる時間にあまり個人差はありません。そこでより実用的な例として、私たち自身の割り当てを考えました。

【例題 3】

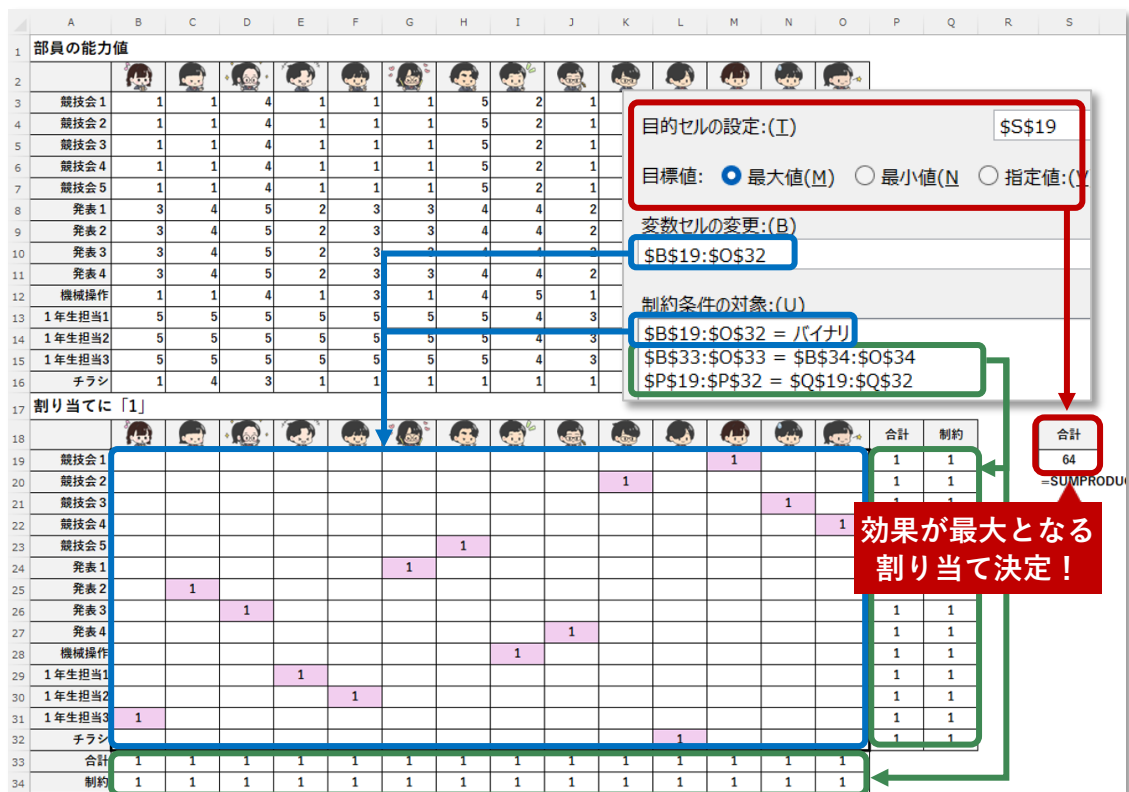
例題

3

鳥栖商業情報処理部は、夏に大きな大会を 2 つ控えています。一つは「情報処理競技大会」、もう一つが「生徒商業研究発表大会」です。他にも生徒会活動や下級生の指導、外部から依頼が来るコンテンツの作成があります。

これまでの実績からそれぞれのスキルをポイント化し、1 人 1 役になるよう割り当てを行います。

【Microsoft Excel によるモデル化とソルバーの実行結果】



実際の割り当て



今回の大会には割り当てられたメンバーで臨んでいます。

従業員のスキルや機械の性能などからも、より最適な割り当てを考えることができ、割り当て問題も現実的な経営活動に置き換えて活用できます。

Lv.Max 巡回セールスマン問題

(1) 巡回セールスマン問題とは

冒頭に述べた今回の研究をはじめのきっかけのエピソードである「長期間欠席した友人が課題の提出をする際、校内各所にいらっしゃる先生方に、どの順番、どの経路でまわった方がよいか」は 0-1 整数計画問題でも代表的な「巡回セールスマン問題」というものに適合します。

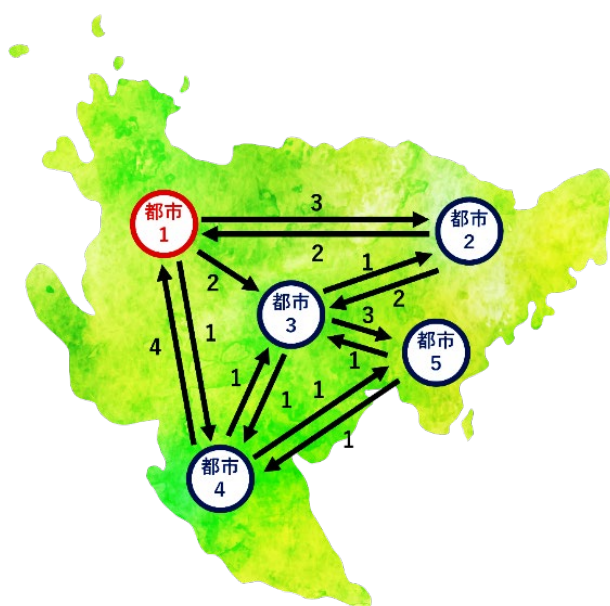
「巡回セールスマン問題」とは、複数の都市のある 1 都市を出発し、すべての都市を 1 度ずつ巡回して最初の都市に戻るとき、最も少ない総移動時間（または距離）となる巡回路を求める問題です。

セールスマンを例とされていますが、この問題は「配送経路の最適化」につながります。私たちの学校がある鳥栖市は九州の東西南北をクロスする流通の拠点であり、その地理的環境から本校には「流通経済科」が設置されています。

また、近年流通業界は、人手不足、コスト上昇、環境への負荷への対応など様々な問題を抱えており、DX (Digital Transformation) の推進やサプライチェーン全体の最適化が求められています。

各流通業者は既に配送経路を最適化するシステムを構築していると考えられますが、流通を学ぶ私たちが流通システムについて研究することは、有意義なものであると考え、挑戦しました。

(2) 巡回セールスマン問題のモデル化



左図のように各都市間の移動にかかる時間を設定したとき、最も少ない時間で全ての都市をまわる「巡回セールスマン問題」を考えます。

数学的に表現するのが難解でしたので、専門書の内容を参考にして Excel でモデル化を行い、ソルバーで最適解を求めました。

『Excel によるシステム最適化』図 1.2 を参考に作成

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
1	決定変数	Y12	Y13	Y14	Y21	Y23	Y32	Y34	Y35	Y41	Y43	Y45	Y53	Y54	x2	x3	x4	x5	制約量	
2	目的関数	3	2	4	2	2	1	1	3	1	1	1	1	1						所要時間
3	制約条件1	1	1	1															1	
4	制約条件2				1	1													1	
5	制約条件3						1	1	1										1	
6	制約条件4									1	1	1							1	
7	制約条件5												1	1					1	
8	制約条件6				1				1										1	
9	制約条件7	1					1												1	
10	制約条件8		1			1					1			1					1	
11	制約条件9			1				1							1				1	
12	制約条件10								1			1							1	
13	制約条件11					5									1	-1			4	
14	制約条件12						5		5						-1	1			4	
15	制約条件13															1	-1		4	
16	制約条件14									5						1		-1	4	
17	制約条件15											5				-1	1		4	
18	制約条件16												5				1	-1	4	
19	制約条件17																	1	4	
20	制約条件18																-1		4	
21	選択する経路に「1」																			
22	Y12																			
23	Y13																			
24	Y14																			
25	Y21																			
26	Y23																			
27	Y32																			
28	Y34																			
29	Y35																			
30	Y41																			
31	Y43																			
32	Y45																			
33	Y53																			
34	Y54																			
35	x2	5																		
36	x3	4																		
37	x4	0																		
38	x5	3																		

目的セルの設定:(I) \$G\$22

目標値: ☐ 最大値(M) ☒ 最小値(N) ☐ 指定値:(V)

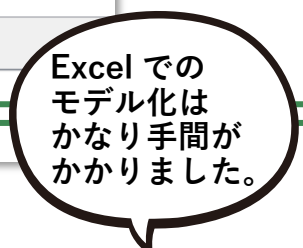
変数セルの変更:(B) \$B\$22:\$B\$38

制約条件の対象:(U)

\$B\$22:\$B\$34 = バイナリ

\$G\$23:\$G\$32 = \$S\$3:\$S\$12

\$G\$33:\$G\$40 <= \$S\$13:\$S\$20



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
1	決定変数	Y12	Y13	Y14	Y21	Y23	Y32	Y34	Y35	Y41	Y43	Y45	Y53	Y54	x2	x3	x4	x5	制約量	
2	目的関数	3	2	4	2	2	1	1	3	1	1	1	1	1						所要時間
3	制約条件1	1	1	1															1	
4	制約条件2				1	1													1	
5	制約条件3						1	1	1										1	
6	制約条件4									1	1	1							1	
7	制約条件5												1	1					1	
8	制約条件6				1				1										1	
9	制約条件7	1					1												1	
10	制約条件8		1			1					1			1					1	
11	制約条件9			1				1							1				1	
12	制約条件10								1			1							1	
13	制約条件11					5									1	-1			4	
14	制約条件12						5		5						-1	1			4	
15	制約条件13															1	-1		4	
16	制約条件14									5						1		-1	4	
17	制約条件15											5				-1	1		4	
18	制約条件16												5				1	-1	4	
19	制約条件17													5		-1		1	4	
20	制約条件18																-1		4	
21	選択する経路に「1」																			
22	Y12	0																		
23	Y13	0																		
24	Y14	1																		
25	Y21	1																		
26	Y23	0																		
27	Y32	1																		
28	Y34	0																		
29	Y35	0																		
30	Y41	0																		
31	Y43	0																		
32	Y45	1																		
33	Y53	1																		
34	Y54	0																		
35	x2	5																		
36	x3	4																		
37	x4	0																		
38	x5	3																		

最短経路は「1 → 4 → 5 → 3 → 2 → 1」で
時間は「9」

(3) 巡回セールスマン問題の Web アプリ

Excel でモデル化してわかったことですが、このモデルではデータの入力の手間が非常にがかかります。また、「**部分経路の排除**」という考え方が難解であり、それを **Excel 上で毎回、制約条件として表すことは現実的ではない**と考えました。

私たち情報処理部は Excel だけではなく、プログラミングも学習しています。（参考として、昨年度には「全商 Web アプリコンテスト」で「実用賞（実質全国 4～5 位）」をいただきました。）この能力を生かし、**巡回セールスマン問題を最適化するオリジナルの Web アプリを自作**しました。なお、アプリ内のマップは無料で使用できる OpenStreetMap を使っており、2017 年の先輩方が「聖地巡礼」の研究をされたときの報告書を見て参考にしました。

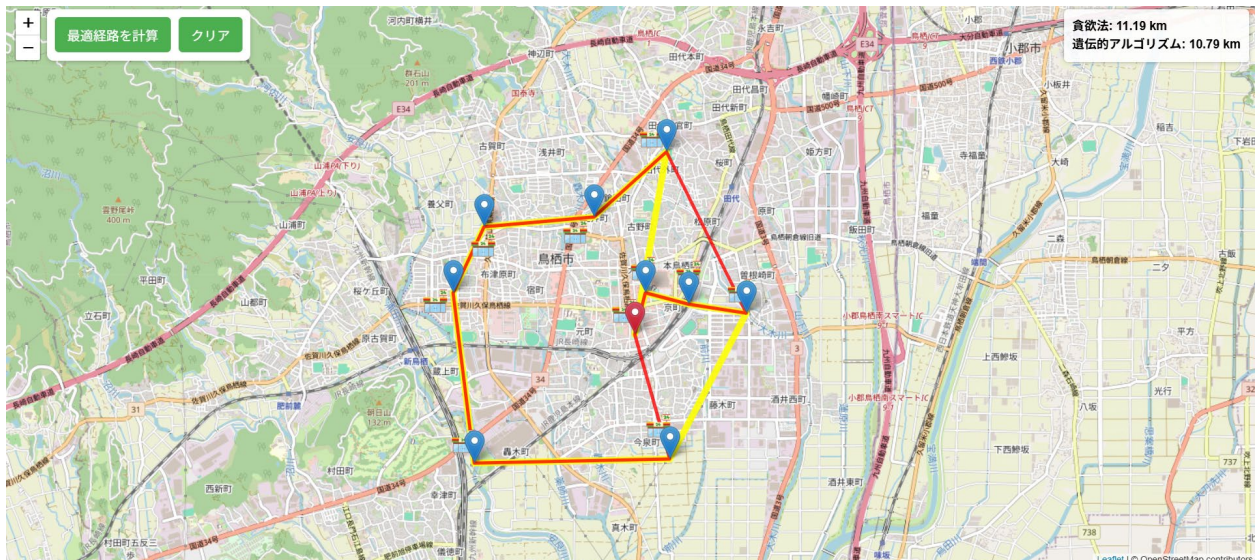
Web アプリはブラウザで動かすことができ、使用機種に依存することなく使用できます。ソースコードは HTML、CSS、JavaScript で入力しています。

< 操作方法 >

1. 地図上の目的地となる各地点をクリック
 - ・ **1 回目のクリック**で赤いマーク表示（スタート地点 かつ ゴール地点）
 - ・ **2 回目以降のクリック**で青いマークが追加表示（経由地）
 - ・ **押し間違えたら右クリック**で取り消し



2. 選択が終わったら、地図左上にある「**最適経路を計算**」ボタンをクリック
3. プログラム内部で最短経路の探索開始！
4. 計算完了→ **2 種類**のルート表示



<コードのポイント>

- 初期座標は J R 鳥栖駅にしていますが、どこにでも設定可能です。
- 座標間の距離は、中学 3 年で習った「三平方の定理」で算出することができます。

```
// 距離を計算する関数
function calculateDistance(pointA, pointB) {
  var dx = pointA.lat - pointB.lat;
  var dy = pointA.lng - pointB.lng;
  return Math.sqrt(dx * dx + dy * dy);
}
```

- 経路が視覚的にわかりやすいように、**地点間に線が引かれる**ようにしています。計算に時間がかかる場合があるので、ユーザーエクスペリエンス (UX) の向上を目的として**リアルタイムで計算中の経路が描画**されます。

```
// 最適経路を描画
function drawRoute() {
  var tspRoute = solveTSP(points);
  var geneticRoute = geneticAlgorithm(points);

  if (tspRoute.length > 0) {
    tspRoute.push(tspRoute[0]); // スタート地点に戻る
    var latLngs = tspRoute.map(p => [p.lat, p.lng]);
    L.polyline(latLngs, { color: 'yellow', weight: 10 }).addTo(map);
    var tspDistance = calculateTotalDistance(tspRoute);
  }

  if (geneticRoute.length > 0) {
    geneticRoute.push(geneticRoute[0]); // スタート地点に戻る
    var latLngs = geneticRoute.map(p => [p.lat, p.lng]);
    L.polyline(latLngs, { color: 'red' }).addTo(map);
    var geneticDistance = calculateTotalDistance(geneticRoute);
  }
}
```

- 初期バージョンで求めることができるのは**直線距離のルート**です。

< 2つのアルゴリズム >

巡回セールスマン問題は、地点数が多い場合、候補となる解の組合せが天文学的数値に増えるため、すべてを網羅して完全な最適解を求めることは難しいです。最適解により近い解を求めるために様々なアルゴリズムが考えられていますが、私たちは2種類を試しました。

1つ目は「貪欲法」で、地図上の**黄色いルート**です。これはスタート地点からの各地点までの距離を計測し、**最も近い地点を選んでいく**という方法です。アルゴリズムとしては単純であり、地点数が多くても短時間で求めることができます。商業科の科目「プログラミング」で学ぶ範囲で対応可能です。ただ、近い方を選んでいくだけなので、地点が多くなるほど最適解を得ることができませんでした。

「貪欲法」のソースコード

```
// TSP最適経路を求める簡易アルゴリズム (貪欲法)
function solveTSP(points) {
  if (points.length < 2) {
    alert('少なくとも2地点を選択してください');
    return [];
  }

  var path = [points[0]]; // 最初の地点からスタート
  var remaining = points.slice(1); // 残り地点

  while (remaining.length > 0) {
    var lastPoint = path[path.length - 1]; // さっきのターンで選んだ地点
    // 残り地点の中の最初の地点を初期値とする
    var nearestPointIndex = 0;
    // 残り地点の中の最初の地点と、さっき選んだ地点からの距離を求め初期値とする
    var nearestDistance = calculateDistance(lastPoint, remaining[0]);
    // 残り地点の全てと、さっき選んだ地点から距離を求め、最小値を探す
    for (var i = 1; i < remaining.length; i++) {
      var distance = calculateDistance(lastPoint, remaining[i]);
      if (distance < nearestDistance) {
        nearestPointIndex = i;
        nearestDistance = distance;
      }
    }
    // 経路に最小値の地点を追加
    path.push(remaining[nearestPointIndex]);
    // 残り地点から最小値の地点を除く
    remaining.splice(nearestPointIndex, 1);
  }

  // 最後にスタート地点に戻る
  path.push(points[0]);
  return path;
}
```

貪欲法では
あまりいい解は
得られませんでした。



2つ目は「遺伝的アルゴリズム」で、地図上の**赤いルート**です。生物学上の**遺伝の交叉**や**突然変異**の考え方を利用し、生み出された経路の良し悪しを判別していくという考え方です。

このアルゴリズムは次図のようにイメージ的には理解できたのですが、コード化するのが難しく、生成AI（Microsoft Copilot）を活用しながら実装しました。

遺伝的アルゴリズムのフローチャートと解の組合せの例



一般社団法人電力土木技術協会「遺伝的アルゴリズム」内の図を参考に作成

「遺伝的アルゴリズム」のソースコード（一部）

```
// 遺伝的アルゴリズムの主要関数
// 初期集団を作成する関数
// size: 集団のサイズ
// points: 巡回する地点のリスト
function createPopulation(size, points) {
  return Array.from({ length: size }, () => points.slice().sort(() => Math.random() - 0.5));
}

// ルートの適応度 (fitness) を計算する関数
// route: 評価対象の経路
// 経路の総距離を求めることで、最適化の指標とする
function fitness(route) {
  let totalDistance = 0;
  for (let i = 0; i < route.length - 1; i++) {
    totalDistance += calculateDistance(route[i], route[i + 1]);
  }
  return totalDistance;
}

// 選択 (selection) 関数
// 集団を適応度の低い順にソートし、最適な個体を半分選ぶ
function selection(population) {
  return population.sort((a, b) => fitness(a) - fitness(b)).slice(0, population.length / 2);
}

// 交叉 (crossover) 関数
// 2つの親個体を組み合わせて子個体を生成
function crossover(parent1, parent2) {
  const start = Math.floor(Math.random() * parent1.length);
  const end = Math.floor(Math.random() * parent1.length);

  // 一部の遺伝情報をコピー
  const child = parent1.slice(Math.min(start, end), Math.max(start, end));

  // 親2の要素を可能な限り追加
  parent2.forEach(city => { if (!child.includes(city)) child.push(city); });

  return child;
}
```

遺伝的アルゴリズムの解は、
貪欲法の解より優秀です！

```
// 突然変異 (mutate) 関数
// route: 変異させる対象の経路
// rate: 突然変異の発生率 (デフォルトは10%)
function mutate(route, rate = 0.1) {
  if (Math.random() < rate) {
    let i = Math.floor(Math.random() * route.length);
    let j = Math.floor(Math.random() * route.length);

    // ランダムに2都市を入れ替え
    [route[i], route[j]] = [route[j], route[i]];
  }
  return route;
}

// 遺伝的アルゴリズムのメイン関数
// points: 最適化したい地点のリスト
// generations: 世代数 (進化の回数)
function geneticAlgorithm(points, generations = 10000, populationSize = 1000) {
  // 初期集団の作成
  let population = createPopulation(populationSize, points);

  for (let i = 0; i < generations; i++) {
    // 選択と突然変異を適用
    population = selection(population).map((route) => mutate(route));

    let newGeneration = [];

    // 交叉による新世代の生成
    for (let j = 0; j < populationSize; j++) {
      let parent1 = population[Math.floor(Math.random() * population.length)];
      let parent2 = population[Math.floor(Math.random() * population.length)];
      newGeneration.push(crossover(parent1, parent2));
    }

    population = newGeneration;
  }

  // 最適な経路を返す
  return selection(population)[0];
}
```

(4) 人間の経験 vs Web アプリ

次にこの Web アプリが示す経路が最適に近いと言えるのか、人間の経験により求められたルートと比較を行いました。検証方法として、鳥栖市内のセブンイレブン 10 店舗に商品を配達するという設定で、地図上の直線距離だけでなく、実際の道路上を車でまわり、距離と時間を実測しました。具体的な実測方法は以下のとおりです。なお、貪欲法では良い解があまり出ないことがテストの段階でわかっていましたので、遺伝的アルゴリズムのルートを用いています。


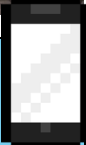




- 「人間の経験」によるルートと「Web アプリ」が示すルートで比較
- 2つのルートを生徒のナビで先生が運転
- 法定速度遵守
- 距離は車のメーターで計測

比較にあたり、人間代表としてご協力いただいたのがスマイルタクシー様（正式社名：(株)鳥栖構内タクシー）です。ドライバーの方に 10 店舗の店名と店舗の写真をお示したところ、なんと地図を見ずにルートを考えられました。

両者のルートは以下のとおりです。



	 人間の経験 スマイルタクシー様	Web アプリ 遺伝的アルゴリズム 
ルート	①中央店→②本鳥栖町店→③大正町店→④田代代官町店→⑤宿町店→⑥養父町店→⑦蔵上 2 丁目店→⑧幸津町店→⑨今泉店→⑩曾根崎町店→①中央店	①中央店→②今泉店→③幸津町店→④蔵上 2 丁目店→⑤養父町店→⑥宿町店→⑦田代大官町店→⑧大正町店→⑨本鳥栖町店→⑩曾根崎町店→①中央店
マップ		

地図上の直線距離と実測した結果は下表のとおりです。なお、スマイルタクシー様にはルートだけ教えてもらい、運転は私たちの案内で顧問の先生が行いました。

	人間の経験 スマイルタクシー様	Web アプリ 遺伝的アルゴリズム
直線距離	11.58km	11.57km
車で実測	16km 36 分 06 秒	17km 45 分 36 秒

直線距離ではアプリのルートの方が 10m だけ短くなりました。しかし、車の実測では距離と時間ともにスマイルタクシー様の圧勝であり、地図を見ずにルート設定ができるプロのドライバーのスキルと経験値の高さを感じる結果となりました。

(5) 原因の分析と改善

改善のために計算量を見直しました。遺伝的アルゴリズムは交叉や突然変異を繰り返して良い解を見つけるため、世代が多いほど、より良い解を見つけることができます。そこで、世代のパラメータを変更し、コンピュータの計算量を増やしました。しかし、新たな良解は生まれなかったため、直線距離としては最適解が既に出ていると思われます。

次に考えたのがルートの向きです。どちらから回っても距離は同じなので特に意識していませんでしたが、アプリのルートを運転中の先生から何度も言われた言葉があります。それは、「また右折か…。」です。実際に右折の数を数えてみたところ 15 か所ありました。対してスマイルタクシー様のルートではたったの 6 か所でした。そこで周る方向を逆向きに変更したところ右折は 8 か所に減りました。また、どの道を通るかについても先生が不満そうな顔をされていたので、私たちのナビをやめて全て先生に決めてもらい、再計測を行いました。その結果が以下のとおりです。

	Web アプリ 遺伝的アルゴリズム (時計回り・生徒のナビ)	Web アプリ 遺伝的アルゴリズム (反時計回り・先生の経験)
車で実測	17km 45 分 36 秒	16.4km 42 分 40 秒

右折で待っている時間は平均 20 秒ほどあり、全体で約 3 分減らすことができました。また、先生の経験に基づく道路変更により距離が 600m 短くなりました。このように人間の経験も取り入れることで、より良い解が得られることがわかりました。

(6) 人件費・燃料費削減のシミュレーション

人件費と燃料費にどれだけ影響するか、先ほどの 3 分減と 600m 減を例とし、以下のように実際の現場を想定した設定でシミュレーションを行ったところ、相当量の削減が期待できることがわかります。

時間 3 分減、距離 600m 減とした場合の人件費と燃料費の変化	
設定	(以下の数値は各種データをもとに計算しやすい数値で設定)
	●配送：中型トラック車 5 台、1 日 10 周、年間 250 日
	●時給：1,500 円（各種求人より）
	●軽油代：150 円/L（地域のガソリンスタンドの価格より）
	●燃費：（各種データより）

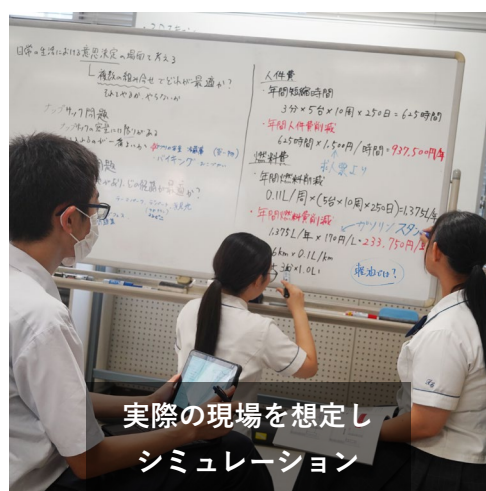
・走行燃費：8km/L \Rightarrow 0.125L/km

・アイドリング中の燃費：1.0 L/時間

※短縮の 3 分は右折待ちの時間のためアイドリング中として計算



地域のガソリン価格を調査



実際の現場を想定し
シミュレーション

【計算結果】

人件費	●年間短縮時間：3 分 \times 5 台 \times 10 周 \times 250 日 = 625 時間
	●年間人件費削減：625 時間 \times 1,500 円/時間 = 937,500 円/年
燃料費	●1 周あたり燃料削減：0.6 km \times 0.125 L/km（距離減の分） + 3 分 \times 1.0 L/時間（アイドリング中分） = 0.075 + 0.05 = 0.125 L/周
	●年間燃料削減：0.125 L/周 \times (5 台 \times 10 周 \times 250 日) = 1,562.5 L/年
	●年間燃料費削減：1,562.5 L \times 150 円/L = 234,375 円/年

5▶ プロジェクトの検証・評価

(1) 仮説の検証

私たちが立てた仮説「日常生活や経営活動の意思決定の場面を 0-1 整数計画問題としてモデル化し、人間が考えた解より優れた解をコンピュータで求めることができる。」に対して、実測した結果、現段階ではまだ不十分という状況ですが、多くの成果があります。

最初の 2 つの問題は 0-1 整数計画問題の入口として取り組み、普段使い慣れている Excel を利用することで、問題の把握、モデル化、解法を理解するのに役立ちました。ただ、頻繁に最適化を行う場合、Excel では手間がかかりすぎるのがわかり、巡回セールスマン問題では手軽に利用できるアプリを完成させることができました。

アプリの制作と実測では、効率的なアルゴリズムを利用するだけでなく、人間の経験も取り入れるとより良い解が出るのが実証できました。また、シミュレーションの結果、短時間・少量の削減であっても、年間を通して換算すると、経営に影響を及ぼす額になることもわかりました。

(2) 流通業者（ヤマト運輸）からの視点より

今回の研究内容を客観的に評価してもらうため、各企業・自治体の皆さまよりご意見を伺っています。最初は流通最大手のヤマト運輸 佐賀支店様です。

- 流通を生業としている企業であるため配達システムは既にある。
- ただ、**当社のシステムと皆さんのアプリは基本的な使い方はほぼ同じ。**
- 流通業界にまだ携わっていない高校生がここまで作っているのに感心した。
- 時間指定などお客様の都合に合わせることができればなお良い。



ドライバー経験のある営業担当者様より高い評価をいただきました。また、会社の効率よりもお客様目線でサービスを考えておられるという点が印象に残りました。

(3) ITシステム開発会社（プロシップ）からの視点より

システム開発の観点からのご助言をいただいています。東京に本社があり、固定資産管理ソフトの開発・販売をされている(株)プロシップ様に私たちの Web アプリを見ていただきました。次のような感想とご助言をいただきました。

- 取組として大変すばらしい。
- 実際の道路のルート設定はかなり難しいと思うがチャレンジしてほしい。
- プロの現場でも生成A Iは当たり前のように使っているし、使えるようになっておいた方がよい。
- ただ、A Iは不十分な部分もあるので、ビジネスで使用する場合は、**A Iが生成したコードをそのまま使うのではなく、人間が手直しする必要がある。**

生成A Iの利用は邪道かもしれないと少し感じていましたが、プロも利用されているということなので、頼り過ぎない程度に利用していこうと思います。ただ、ご助言にもあったとおり、A Iのミスや、こちらの指示が伝わっていないことがアプリ制作中に何度も発生しました。適度に参考として使いながら、自分たち自身のスキルを伸ばしていきたいと思っています。

(4) 流通業者以外（J T B・J R九州・自治体等）の視点より

流通の視点で研究を行ってきましたが、ルート設定が必要な場面は他にも多くあります。**他業種でも利用価値**があると考え、多くの皆さまに Web アプリを紹介するとともにご意見を伺いました。

最初に**観光**での利用方法を探るために**J T B佐賀支店様**に紹介し、次のような感想とご助言をいただきました。

- 高校生でこのようなアプリを作っていることに驚いている。ぜひ使ってみたい。
- 観光に使うのであれば、施設の利用可能時間、自由時間の長さ、トイレ休憩、お土産を買う時間など、**施設やお客様のご都合**を考慮する必要がある。
- 旅行、観光でも使えるよう開発を継続してほしい。



J T Bの皆さま

また、本校の学校運営協議会（J R九州、佐賀新聞社、地域住民・自治体の皆さま）で紹介し、それぞれのお立場より次のようなご意見をいただきました。



J R九州 駅長様

- J R九州駅長様より…駅を基点に観光に行くときの「**二次交通**」のルート設定に役立つと思う。観光名所や飲食店も紹介してもらえれば。
- 佐賀新聞社様より…自治体が「**乗合タクシー**」のサービスを始めるという情報がある。利用者の時間に合わせて回れるような仕組みがあるとよいのでは。

ここでもお客様、地域住民の方の利便性を第一に考えられた意見が多く、流通やサービス業で大切なことを気づかされました。

(5) 成果・課題・改善と展望

< 成果 >

Web アプリという成果物を作り出すことができたという点が大きいです。最適解ではないかもしれませんが、「良い解」は出せていると思います。実用例として先生方が中学校や企業回りをされる際のルート設定に役立っています。

参考までに、もし日本橋から出発して都内の主要な【神社仏閣パワースポット巡り】をする場合は、

①日本橋→②神田明神→③浅草寺→④根津神社→⑤明治神宮→⑥日枝神社→⑦増上寺→①日本橋 が直線距離的にはお勧めです。



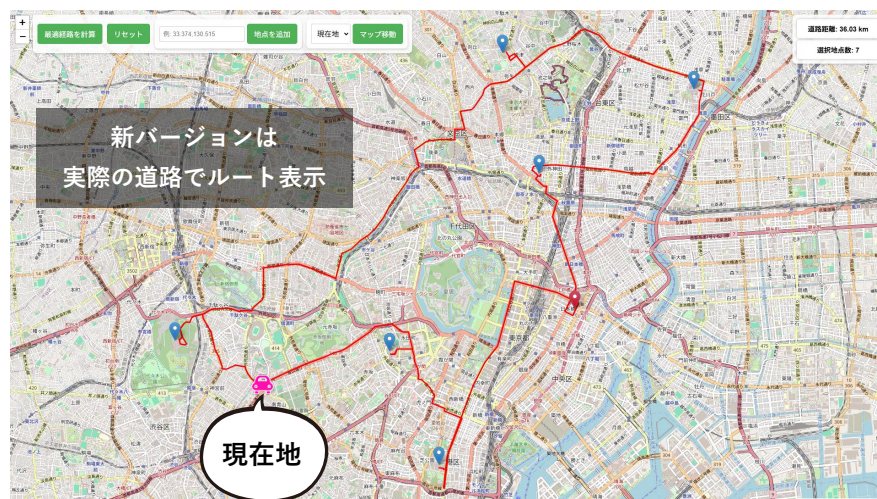
< 課題 >

私たちの Web アプリは直線距離のデータをもとにルートを求めますが、実測した結果、直線距離で計測したルートでは最適解とは限らないことがわかりました。スマイルタクシー様は、ルート選定において単に道のりだけでなく、運転時の難易度や道路状況まで考慮されていました。右折は全て時間がかかるのではなく、交通の流れがスムーズな時ほど難しく、渋滞時の方が進入しやすいという点、トラックは高さ制限のある場所は避ける点などです。

運転経験のない私たちには想像すらできなかった人間の経験値をデータとして取り入れることができれば、より精度の高い最適化ができる予想されます。また、私たちの Web アプリに不足しているのは、「相手の都合」です。配達では「時間指定」、観光では「利用可能時間」などを考慮する必要があります。アプリの利用者だけでなく、関係者全体の利便性を第一に考えて設計する必要があると再認識しました。

< 改善と展望 >

アプリはバージョンアップを重ね、現在は見た目のルートを直線距離ではなく実際の道路に沿って表示するように改善しました。また、GPSによる現在地の表示、各種入力ボタンなど、ユーザの利便性を高める機能を追加しています。



実際の道路データは現バージョンではルート表示だけに使用していますが、これを**計算上のデータ**として扱えるようにしていきます。また、道路の混雑状況や、相手の都合、人間の経験を取り入れ、短時間で精度の高い経路を導き出せるよう、様々な**データの設定方法や最適化のアルゴリズム**をさらに学習していきます。

研究の過程では**新たなアイデア**も生まれています。ルート設定の用途を特化し、**ニッチなアプリ**に挑戦したいです。考えているアイデアは以下のとおりです。

- 鳥栖周辺の駅からの二次交通ルート ●乗合タクシーのルート ●不動産探し
- 宅配弁当・訪問介護ルート用 ●高齢者向けショッピングセンター等の買い物ルート

6 ▶ おわりに

おわりにあたって、プロジェクトメンバーの感想を述べます。

問題集の問題でしかソルバーを使ったことがありませんでしたが、新しい応用例に触れることができました。この研究を通じて社会に貢献していきたいです。



今回は数理最適化というかなり難しい研究でしたが日常の意思決定の場面でソルバーやアプリを使って効率よく解決することができ、満足しています。



授業で習う数学の問題よりもっと様々な活用方法があることに驚きました。数学は得意ではないですが、将来役立つことを意識してもっと勉強したいです。



私は地図を見ることが好きなので、今回の研究は興味深く取り組むことができました。タクシードライバーの方のご助言はとても良い刺激になりました。



生産性を上げるためにAIを使いながらWebアプリを制作しました。ただAIも完璧ではないので、1行ずつ理解しながら、形にすることができました。



(株)鳥栖構内タクシー（スマイルタクシー）様と

<参考書籍>

- ▶岡部恒治ほか『改訂版 高等学校 数学 A』数研出版
- ▶植竹朋文ほか『ソフトウェア活用』東京法令出版
- ▶久保幹雄『組合せ最適化とアルゴリズム』共立出版
- ▶大野勝久・田村隆善・伊藤崇博『Excel によるシステム最適化』コロナ社
- ▶中山舜民・橘海里・オフィス sawa『マンガでわかる数理最適化』オーム社

<参考Webサイト>

- ▶スタビジ「【誰でも AI データサイエンス】 by ウマたん」<https://www.youtube.com/@aiby8596>
- ▶ヴェイパー・エイオン「【おもしろ技術】遺伝的アルゴリズムとは？【解説】」
<https://www.youtube.com/watch?v=-bdgUwGA2AE>
- ▶一般社団法人電力土木技術協会「遺伝的アルゴリズム」
https://www.jepoc.or.jp/tecinfo/library.php?_w=Library&_x=detail&library_id=53
- ▶LOGI JOURNAL 「物流業界の人手不足の原因とは？深刻化の背景と今後の解決策を解説」
<https://www.plc.co.jp/blog/lack-of-manpower/>
- ▶Microsoft Copilot <https://copilot.microsoft.com/>
- ▶フキダシデザイン <https://fukidesign.com/>
- ▶Powered Template <https://poweredtemplate.com/>
- ▶CHARAT <https://charat.me/>
- ▶Hatenabase 生成 AI が切り拓く運輸・物流業界の未来：国内外の革新事例と展望
<https://hatenabase.jp/blog/>
- ▶OpenStreetMap <https://www.openstreetmap.org/>



(株)ヤマト運輸佐賀主管支店・鳥栖蔵上営業所の皆さまと