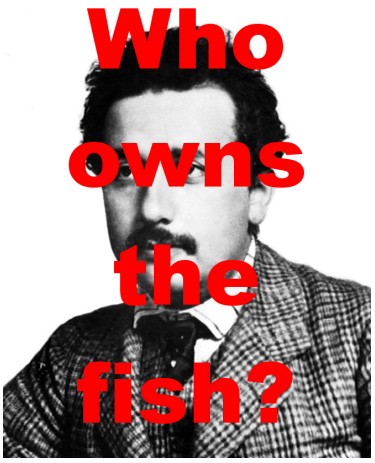
Solving the Einstein Riddle

Curtis Bright, Maplesoft

- The "Einstein Riddle" is a logic puzzle apocryphally attributed to Albert Einstein and is often stated with the remark that it is only solvable by 2% of the world's population. The true source of the puzzle is unknown, but a version of it appeared in the magazine *Life International* in 1962.
- The puzzle is stated as follows:

There are five houses in a row with each house a different colour and each house owned by a man of a different nationality. Additionally, each of the owners have a different pet, prefer a different kind of drink, and smoke a different brand of cigarette. Furthermore, we know the following information:

1. The Brit lives in the Red house.
2. The Swede keeps dogs as pets.
3. The Dane drinks tea.
4. The Green house is next to the White house, on the left.
5. The owner of the Green house drinks coffee.
6. The person who smokes Pall Mall rears birds.
7. The owner of the Yellow house smokes Dunhill.
8. The man living in the centre house drinks milk.
9. The Norwegian lives in the first house.
10. The man who smokes Blends lives next to the one who keeps cats.
11. The man who keeps horses lives next to the man who smokes Dunhill.
12. The man who smokes Blue Master drinks beer.
13. The German smokes Prince.
14. The Norwegian lives next to the Blue house.
15. The man who smokes Blends has a neighbour who drinks water.



The puzzle is: **Who owns the fish?**

- We can solve this puzzle using Maple's built-in efficient SAT solver. A SAT solver takes as input a formula in Boolean logic and returns an assignment to the variables which makes the formula true (if one exists). See the Satisfy command for more information.

Setting up the problem

- We will label the houses with a number i between 1 and 5 and use the Boolean variables $S_{i,\,attribute}$ where $1 \le i \le 5$ and *attribute* is either a colour, a nationality, a pet, a drink, or a cigarette brand.
- The variable $S_{i,\,attribute}$ will be assigned true when the ith house or its owner is the person who has the attribute given by *attribute*.
- We will use the following sets to define the attributes and house indices:

```
houses := {1, 2, 3, 4, 5}:

colours := {blue, green, red, white, yellow}:
nationalities := {Brit, Dane, German, Norwegian, Swede}:
drinks := {beer, coffee, milk, tea, water}:
cigarettes := {Blends, BlueMaster, Dunhill, PallMall, Prince}:
pets := {birds, cats, dogs, horses, fish}:

attributeTypes := [colours, nationalities, drinks, cigarettes,
pets]:
allAttributes := `union`(attributeTypes[]):
```

Generating the constraints

- We know that each attribute is not shared among the five houses or their owners. Since there are exactly five houses, each attribute must appear exactly once among the five houses.
- The knowledge that each attribute appears at least once can be encoded as the clauses

$S_{1,a} \vee S_{2,a} \vee S_{3,a} \vee S_{4,a} \vee S_{5,a}$ for each attribute a.

```
allAttributesAppear := seq(&or(seq(S[i, a], i=1..5)), a in allAttributes):
```

- The knowledge that each attribute is not shared can be encoded as the clauses $S_{i,a} \Rightarrow \neg S_{j,a}$ where j is a house index not equal to i and a is an attribute (equivalently, $\neg S_{i,a} \vee \neg S_{j,a}$).

```
noAttributesShared := seq(seq(seq(S[i, a] &implies &not(S[j, a]), j=i+1..5), i=1..5), a in allAttributes):
```

- Additionally, we know that each house has some colour, which is encoded as $S_{i,blue} \vee S_{i,green} \vee S_{i,red} \vee S_{i,white} \vee S_{i,yellow}$ for each house i (and similarly for the other attribute types).

```
allHaveEachAttribute := seq(seq(&or(seq(S[i, a], a in
attributeTypes[j])), i=1..5), j=1..5):
```

- The knowledge that each house cannot have two colours can be encoded as the clauses $S_{i,\,c} \Rightarrow \neg S_{i,\,d}$ where c and d are two distinct colours (and similarly for the other attribute types).

```
noneHaveMultiple := seq(seq(seq(seq(S[i, attributeTypes[j][k]]
&implies &not(S[i, attributeTypes[j][l]]), l=k+1..5), k=1..5),
j=1..5), i=1..5):
```

Translating the known facts into logic

- We now translate each of the 15 pieces of information into logical clauses and store the clauses in the array *clues*.

```
clues := Array(1..15):
```

1. The Brit lives in the Red house. This has the form $S_{i,\,Brit} \Rightarrow S_{i,\,red}$ for house indices i.

```
clues[1] := seq(S[i, Brit] &implies S[i, red], i=1..5):
```

2. The Swede keeps dogs as pets. This has the form $S_{i,\,Swede} \Rightarrow S_{i,\,dogs}$ for house indices i.

```
clues[2] := seq(S[i, Swede] &implies S[i, dogs], i=1..5):
```

3. The Dane drinks tea. This has the form $S_{i,\,Dane} \Rightarrow S_{i,\,tea}$ for house indices i.

```
clues[3] := seq(S[i, Dane] &implies S[i, tea], i=1..5):
```

4. The Green house is next to the White house, on the left. This has the form $S_{i,\,green} \Rightarrow S_{i+1,\,white}$ for house indices $i < 5$ as well as $\neg\, S_{5,\,green}$ since the green house cannot be the leftmost house.

```
clues[4] := seq(S[i, green] &implies S[i+1, white], i=1..4), &
not(S[5, green]):
```

5. The owner of the Green house drinks coffee. This has the form $S_{i,green} \Rightarrow S_{i,\ coffee}$ for house indices i.

```
clues[5] := seq(S[i, green] &implies S[i, coffee], i=1..5):
```

6. The person who smokes Pall Mall rears birds. This has the form $S_{i,\ PallMall} \Rightarrow S_{i,\ birds}$ for house indices i.

```
clues[6] := seq(S[i, PallMall] &implies S[i, birds], i=1..5):
```

7. The owner of the Yellow house smokes Dunhill. This has the form $S_{i,\,yellow} \Rightarrow S_{i,\,Dunhill}$ for house indices i.

```
clues[7] := seq(S[i, yellow] &implies S[i, Dunhill], i=1..5):
```

8. The man living in the centre house drinks milk. This has the form $S_{3,\,milk}$.

```
clues[8] := S[3, milk]:
```

9. The Norwegian lives in the first house. This has the form $S_{1,\,Norwegian}$.

```
clues[9] := S[1, Norwegian]:
```

10. The man who smokes Blends lives next to the one who keeps cats. This has the form $S_{i,Blends} \Rightarrow \left(S_{i-1,\,cats} \vee S_{i+1,\,cats}\right)$ for house indices $1 < i < 5$, $S_{i,\,Blends} \Rightarrow S_{i+1,\,cats}$ for $i = 1$, and $S_{i,Blends} \Rightarrow S_{i-1,\,cats}$ for $i = 5$.

```
clues[10] := seq(S[i, Blends] &implies (S[i-1, cats] &or S[i+1, cats]), i=2..4), S[1, Blends] &implies S[2, cats], S[5, Blends] &implies S[4, cats]:
```

11. The man who keeps horses lives next to the man who smokes Dunhill. This has the form $S_{i,\,horses} \Rightarrow \left(S_{i-1,\,Dunhill} \vee S_{i+1,\,Dunhill}\right)$ for house indices $1 < i < 5$, $S_{i,horses} \Rightarrow S_{i+1,Dunhill}$ for $i = 1$, and $S_{i,horses} \Rightarrow S_{i-1,Dunhill}$ for $i = 5$.

```
clues[11] := seq(S[i, horses] &implies (S[i-1, Dunhill] &or S[i+1, Dunhill]), i=2..4), S[1, horses] &implies S[2, Dunhill], S[5, horses] &implies S[4, Dunhill]:
```

12. The man who smokes Blue Master drinks beer. This has the form $S_{i,\,BlueMaster} \Rightarrow S_{i,\,beer}$ for house indices i.

```
clues[12] := seq(S[i, BlueMaster] &implies S[i, beer], i=1..5)
:
```

13. The German smokes Prince. This has the form $S_{i,\,German} \Rightarrow S_{i,\,Prince}$ for house indices i.

```
clues[13] := seq(S[i, German] &implies S[i, Prince], i=1..5):
```

14. The Norwegian lives next to the Blue house. This has the form $S_{i,\,Norwegian} \Rightarrow \left(S_{i-1,\,blue} \vee S_{i+1,\,blue}\right)$ for house indices $1 < i < 5$, $S_{i,Norwegian} \Rightarrow S_{i+1,blue}$ for $i = 1$, and $S_{i,Norwegian} \Rightarrow S_{i-1,\,blue}$ for $i = 5$.

```
clues[14] := seq(S[i, Norwegian] &implies (S[i-1, blue] &or S[i+1, blue]), i=2..4), S[1, Norwegian] &implies S[2, blue], S[5, Norwegian] &implies S[4, blue]:
```

15. The man who smokes Blends has a neighbour who drinks water. This has the form $S_{i,\,Blends} \Rightarrow \left(S_{i-1,\,water} \vee S_{i+1,\,water}\right)$ for house indices $1 < i < 5$, $S_{i,Blends} \Rightarrow S_{i+1,water}$ for $i = 1$, and $S_{i,Blends} \Rightarrow S_{i-1,water}$ for $i = 5$.

```
clues[15] := seq(S[i, Blends] &implies (S[i-1, water] &or S[i+1, water]), i=2..4), S[1, Blends] &implies S[2, water], S[5, Blends] &implies S[4, water]:
```

Finding a solution

- We use the Satisfy command from the Logic package which finds a satisfying assignment of a logical formula if one exists.
- We use the Usage command from the CodeTools package to measure how quickly the solution is found.

```
allConstraints := allAttributesAppear, noAttributesShared,
allHaveEachAttribute, noneHaveMultiple, entries(clues, nolist)
:
satisfyingAssignment := CodeTools:-Usage(Logic:-Satisfy(&and(
allConstraints))):
```

- To give the answer to the puzzle we need to determine who owns the fish.

```
for eq in satisfyingAssignment do
     if rhs(eq) and op(2, lhs(eq)) = fish then
          fishHouseIndex := op(1, lhs(eq));
     end if;
end do:

for eq in satisfyingAssignment do
     if rhs(eq) and op(1, lhs(eq)) = fishHouseIndex and op(2,
lhs(eq)) in nationalities then
          fishOwner := op(2, lhs(eq));
     end if;
```

Visualizing the solution

- To easily see who lives where and the attributes of each house and owner we fill a 2D array with data from the found solution:

```
data := Array(1..5, 1..5):

for i from 1 to 5 do
     for j from 1 to 5 do
          for eq in satisfyingAssignment do
               if rhs(eq) and op(1, lhs(eq)) = j and op(2, lhs
(eq)) in attributeTypes[i] then
                    data[i, j] := op(2, lhs(eq));
               end if;
          end do:
     end do:
```