The Tuckerman Traverse

Teacher Notes to support Tenderfoot Unit 5: Theoretical Computers – Fun with finite-state machines

A practical investigation, exploring the properties of a hexahexaflexagon. Their behaviour is captured in a State Diagram and through this the notion of a Finite-State Machine is introduced.

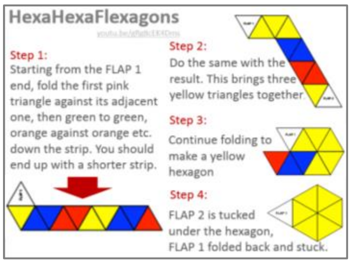
Preparation required:
Part complete class set of A3 hexahexaflexagons (trimmed and folded in half) with assembly instructions. A Hexahexaflexagon Exploration Sheet per student.

Models Of Computation

Textbooks often explain computers by reference to Input/Process/Output - a general model of how something is computed. Computations can be expressed as algorithms – the steps required to accomplish a particular task. Tenderfoot Unit 5 introduces theoretical models of computation that formalise the notion of an algorithm. This may seem a bit obscure, but finding ways of expressing algorithms leads to some very big questions. Using models of computation, famous computer scientists have shown that there are things computers will never be able to solve, no matter how quick or powerful.

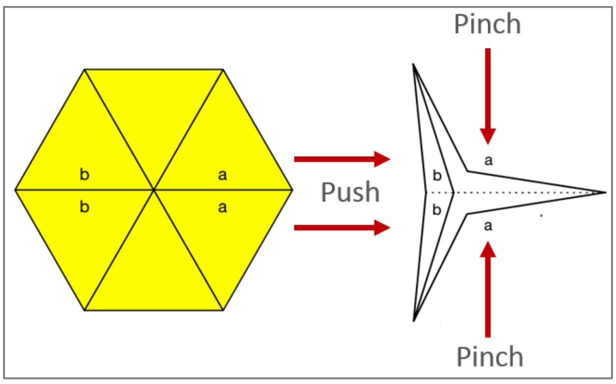
Hexahexaflexagons

Based on material by Paul Curzon (cs4fn), this activity explores the properties of a hexahexaflexagon and uses them to introduce 'finite-state machines'. The supporting booklet is included in the resources and at www.cs4fn.org/hexahexaflexagon/. A hexahexaflexagon is a curious hexagonal shape, made by folding a piece of paper. Their discovery is credited to British mathematician, Arthur H Stone, studying at Princeton University in 1939. Finding his English sized paper didn't fit his American folder, he tore a strip off and folded it up. The Princeton Flexagon Committee was formed with friends, Bryant Tuckerman, Richard Feynman and instructor John Tukey to explore their properties. Some years later (1956), mathematician Martin Gardner popularised them in the magazine Scientific American. In 2012, to celebrate Martin Gardner's birth, on 21 October, Vi Hart produced 3 wonderful videos telling the story of the hexaflexagon. They provide an excellent introduction to the Tuckerman Traverse: youtu.be/VIVIegSt81k. Can children figure out a way to cycle through and display all the faces of a hexahexaflexagon, returning to their starting point?



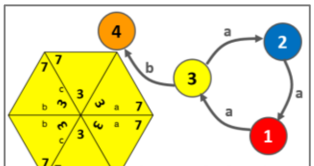
Resources include a template to make hexahexaflexagons. It should be enlarged by 141% to fit A3 paper. Small hexahexaflexagons are hard to manipulate. The shape is best provided cut out, folded along its length and stuck back to back, as accurate folding and sticking is essential. Students can work individually or in groups depending on the number available. The 4 steps on the handout show how to complete it. If students struggle, encourage them to use the video link on the handout.

Flexing is a matter of pinching and flattening the opposite side. The letters and numbers on the flexagon help the student's exploration. Start with the yellow side facing you. If assembled correctly, the 3's should be in a central ring and the lower case a's and b's together. *Always keep the flexagon facing the same way up*. By pinching and pushing, it turns inside out. The pinch points are indicated by adjacent pairs of lower case letters (a, b or c). Encourage initial exploration to see what can be discovered.



The Tuckerman Traverse

After a few minutes exploration, introduce the Tuckerman Traverse. Discussion prompts are given in the slide notes. How many faces does it have (9)? Some faces have the same colour, but can be seen to be different by the positioning of the digits. <u>*With the flexagon kept the same way up*</u>, each face is identified by the central ring of digits. How can we be sure we have explored them all? A diagram is an example of abstraction – removing details that obscure understanding. Which details are important? The 3's in the middle identify the face, so we can draw a node. When we pinch at 'a', we move to face 2. From face 2 we can't pinch anywhere and move back so the arrow is one way. The faces are indicated by nodes, and the transitions indicated by edges linking different faces. Diagrams like these are known as directed graphs (digraphs). The graph represents the transitions between faces. The edges have a letter assigned, indicating where to pinch, and the arrow indicates a direction.

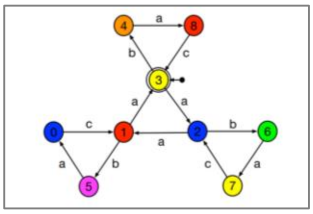

The start of a digraph

Working in small groups, students complete the table to record results. Exploration needs to be systematic. The presentation records the first few moves together. Read the slide notes carefully to ensure you understand the animation. Make sure students complete the table and add the transition to a digraph in the box above.

Part complete exploration table

Node	a	b	c	Fully explored
3	✓	✓	na	✓
2	✓		na	
1	✓		na	

The completed graph models the hexahexaflexagon. It is now easy to plan a 'Tuckerman Traverse'. By abstracting away unnecessary detail the problem is easier to solve. Indicating where to start (with an arrow), students can trace the inputs required to move between different states. A double ring denotes an end state. A graph indicating a set of states and inputs required for each transition is known as a State Diagram. It expresses the behaviour of a 'finite-state machine'.



The graph denotes a finite number of states (nodes). It also indicates the actions (or inputs) required to move from one state to another (the letter to pinch). All possible actions (in this case the letters a, b and c) are known as the machines alphabet – the only acceptable inputs. State Diagrams can also output things. To keep it simple, the only output in this case is displaying the colour and number of the face. State Diagrams are extremely useful. They are a visual representation of a potential sequence of inputs/actions and can therefore be used to model many computational processes.

There is a more difficult extension. Turning the flexagon over reveals a face with no numbers in the middle. There are many more faces to explore flexing it this way up. Is a full traversal of all sides possible? To answer that, we need to decompose the problem into smaller explorations. Decomposing a complex task into smaller parts is a key concept in computational thinking. Suggesting what to investigate first makes a good discussion. Slide notes provide prompts but it is left as an open ended extension.

A simple utility for children to create their own hexahexaflexagon, offers a practical element to end the activity. The result is a vector graphic which can be scaled to maximise the print area available.

State Diagrams, as we have seen, can visually represent the behaviour of something that responds to input, has a set (finite) number of states, and can (if needed) output something too. Artefacts displaying such behaviour are all around us. In Computer Science we call them finite-state machines. Finite-state machines are extremely useful. They can be used to model many (but not all) computational processes. The presentation considers the behaviour of a ball point pen and a combination lock. Traffic lights are a more complex example. A homework exercise might be to draw a state diagram for a pelican crossing, or identify other household objects that can be modelled through state diagrams. Lots of simple electrical or mechanical devices will fit this description.