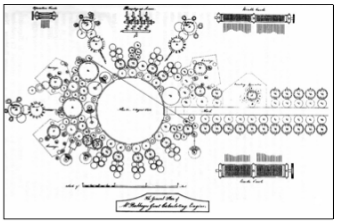
Eric Roberts
CS 54N

Handout #4
October 5, 2016

The Analytical Engine

The Analytical Engine



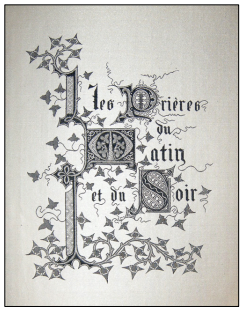
Eric Roberts
CS 54N
October 5, 2016

The Analytical Engine

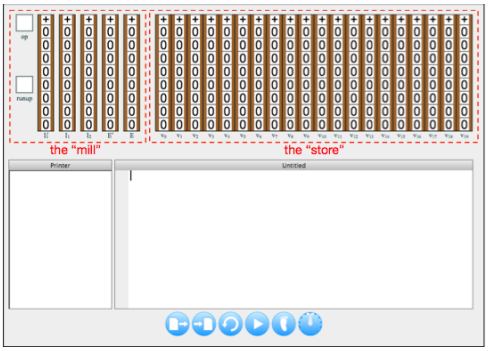
- As Babbage built prototypes of his Difference Engine, he began to envision a much more powerful computing device he called the Analytical Engine.
- Babbage's initial notes on the Analytical Engine appear in 1837, but the most complete description appears in a 1842 paper by Luigi Federico Menabrea, who was reporting on a lecture Babbage gave in 1840. Ada Lovelace translated Menabrea's paper from French into English and provided notes that were three times longer than the original.
- The essential difference between the Difference Engine and the Analytical Engine is that the Analytical Engine was designed to be ***programmable,*** allowing users to perform any sequence of calculations. The programs were encoded on punched cards in the manner of the Jacquard loom, which Ada and her mother had seen in their visits to the English industrial areas.

Jacquard Loom

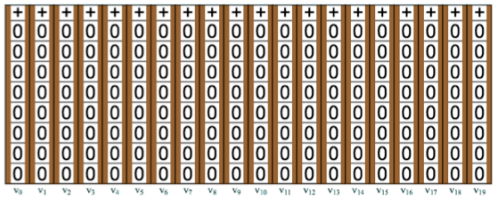
Products of the Jacquard Loom



Structure of the Analytical Engine

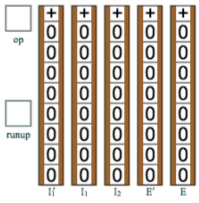


The "Store"



- Each column holds a single integer as in the Difference Engine.
- Numbers in the Analytical Engine are signed.
- Each column has a numeric address: v_0, v_1, v_2, v_3, and so on.

The "Mill"



- The **op** indicator holds the current operation (+, −, ×, ÷)
- The mill has five columns:
  - I_1 and I_2 are the input values
  - E is the output value
  - I_1' and E′ are used to store extra digits for the × and ÷ operations
- The **runup** indicator is set when the result of an operation changes sign.

Instructions for the Analytical Engine

N *address value*	Store *value* in *address*
+	Set the machine to addition
−	Set the machine to subtraction
×	Set the machine to multiplication
÷	Set the machine to division
L *address*	Load from *address*, preserving data
Z *address*	Load from *address*, clearing data
S *address*	Store egress register in *address*
P *address*	Print value in *address*

Program to Add Two Numbers

```
N 0 25   /* First number is in v0        */
N 1 17   /* Second number is in v1       */

+        /* Set machine for addition    */
L 0      /* Load first number into I1   */
L 1      /* A second load does the add  */
S 2      /* Store result in v2          */
P 2      /* Print the result            */
```

Adding Control Operations

N *address value*	Store *value* in *address*
+	Set the machine to addition
−	Set the machine to subtraction
×	Set the machine to multiplication
÷	Set the machine to division
L *address*	Load from *address*, preserving data
Z *address*	Load from *address*, clearing data
S *address*	Store egress register in *address*
P *address*	Print value in *address*
B *number*	Move backward specified number of cards
F *number*	Move forward specified number of cards
?B *number*	Move backward if **runup** lever is set
?F *number*	Move forward if **runup** lever is set

Exercise: Produce a Table of Squares

Use the Analytical Engine to produce a table of squares.

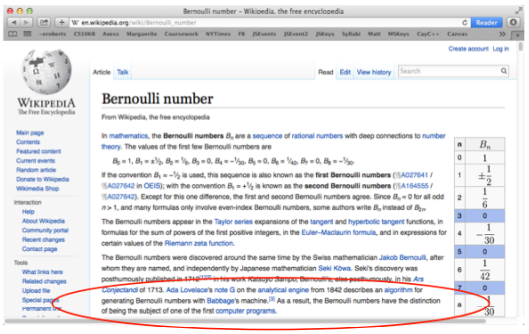
Printer
0
1
4
9
16
25
49
64
81

The simplest approach is to simulate the operation used by the Difference Engine to accomplish the same task.

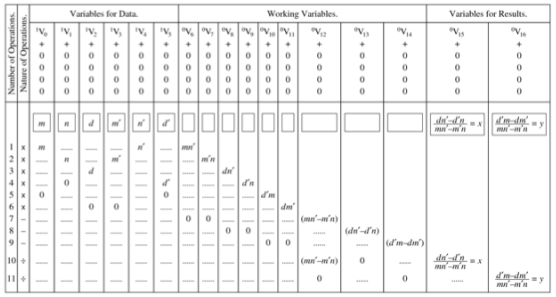
Multiplication and Division

- Babbage recognized that multiplying two integers produces a result that typically has twice the number of digits that appear in the original values.
- To take account of this fact, the multiplication operation for the Analytical Engine produces its result in a pair of columns. Column E shows the ***low-order digits*** of the product, which are the digits on the right that include the units value. Column E′ shows the ***high-order digits***.
- The division operation uses two columns to store the dividend. You set up the division by loading the low-order digits into column I_1 and the high-order digits (if any) into column I_1'.
- As long as you know that the numbers won't exceed the number of digits in a column, you can ignore E′ and I_1' altogether.

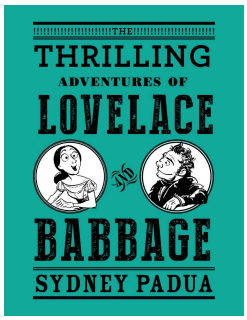
Bernoulli Numbers



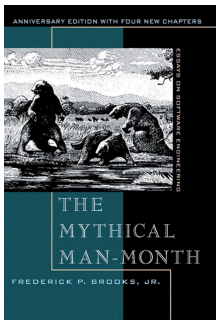
Ada's Program for Bernoulli Numbers



How the Analytical Engine Worked



The Mythical Man-Month



11. Plan to Throw One Away

In most projects, the first system built is barely usable. It may be too slow, too big, awkward to use, or all three. There is no alternative but to start again, smarting but smarter, and build a redesigned version in which these problems are solved. . . .

The management question, therefore, is not *whether* to build a pilot system and throw it away. You *will* do that. The only question is whether to plan in advance to build a throwaway, or to promise to deliver the throwaway to customers.