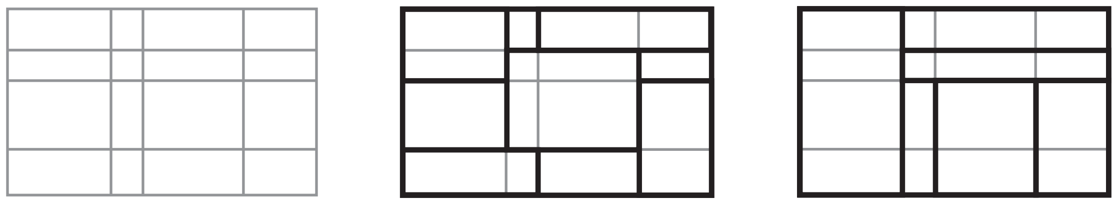
Arranging Rectangles

Hugh Dubberly and Patch Kessler[1]

January 25, 2009

Most documents consist of rectangular blocks of media, such as images, headers, paragraphs, and blank space. An inconsistent arrangement of these blocks makes a document hard to navigate, however no variation is boring. A balance can be achieved by arranging blocks against an underlying rectilinear grid.



Underlying Grid   Arrangement 1   Arrangement 2

A grid is a tool for creating a visual *theme and variations*- a minimum of rules to create a maximum of diversity (or freedom or choice). The idea of unity in diversity is important in many creative endeavors- from architecture to music.

Grids have been described in the design literature for at least 50 years. French architect Le Corbusier describes a grid system in his 1946 book *Le Modulor* [2]. Swiss graphic designer Karl Gerstner describes a number of grid systems or "programmes" in his 1964 book *Designing Programmes* [1]. One of Gerstner's examples is a 15th century system for varying cathedral windows. Use of tatami mats appears to go back 1300 years. Writing systems (both alphabetic and ideographic) are even older examples. The classic work on grids is Josef Muller-Brockman's 1981 *Grid Systems* [3], which documents grid systems in his own work and that of other Swiss designers of the "modern" period. Recently, grids have again become a popular subject of design discussions, especially as a new generation of designers tackles the challenges of designing for the web. See for example, `http://www.thegridsystem.org/`

While much design literature concerns grids, to our knowledge no one has worked out and published a rigorous system for generating variations. That means that claims of minimum rules and maximum variation have not been quantified. In this paper we explore the mathematical underpinnings of grid generation. We present a simple algorithm for generating all rectangular arrangements on a given grid, as well as an expression for the total number of possibilities (see the text following equation 1). This number is enormous even for simple grids- there are 70878 different arrangements on the 4×4 grid shown above.

We introduce a parent/child relationship between arrangements which links all arrangements together in one big network. The number of immediate network relatives of any arrangement is small, and it takes only a few jumps between immediate relatives to get from one arrangement to any other. We do this using interactive Java applets created with Processing (`http://www.processing.org/`). Our applet for navigating 4×4 arrangements is posted online at `http://www.mechanicaldust.com/applet/index.html`.

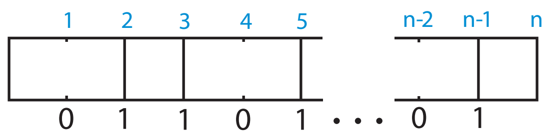
This work has implications for automating the generation of design variations. We imagine off-loading the labor of creating variations to a machine, thus enabling designers to focus on setting up rules and selecting options. We imagine giving a program images and text, and being given back all possible ways of arranging everything on a given grid.

After writing up these notes, we discovered a very similar work [4] by mathematicians J. Smith and H. Verrill, which contains a way of generating variations as well as our formula for the total number of possibilities.

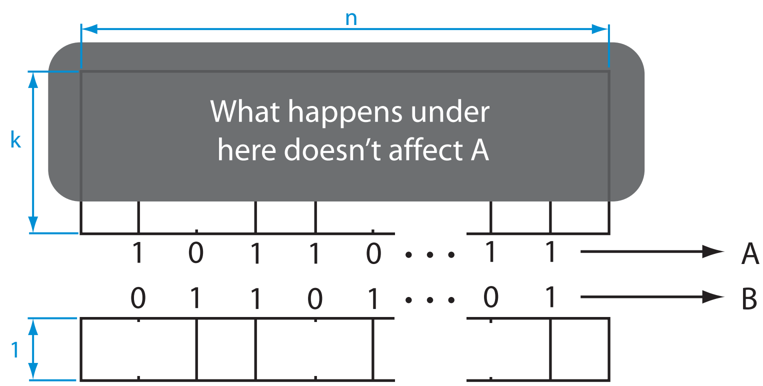
[1] Hugh Dubberly: `hugh@dubberly.com` `http://www.dubberly.com`
Patch Kessler: `patchkessler@gmail.com` `http://www.mechanicaldust.com`

Problem Statement
An i-rectangle is a rectangle with integer height and integer width. Our problem is to find every collection of non-overlapping i-rectangles with an $m \times n$ rectangular union (where m and n are integers).

Generating Arrangements
We generate arrangements by induction, starting with a $1 \times n$ rectangle. Each arrangement of blocks within the $1 \times n$ rectangle corresponds to a different length $n-1$ list of 1's and 0's; each of these lists is the binary expansion of a non-negative integer less than 2^{n-1}.



We use the $k \times n$ arrangements to build the $(k+1) \times n$ arrangements. The edges at the bottom of each $k \times n$ arrangement correspond to an integer A.



Each $k \times n$ arrangement corresponding to A leads to $2^{R_{AB}}$ different $(k+1) \times n$ arrangements corresponding to B, where R_{AB} is an integer which depends on how well the binary expansions of A and B coincide. As an equation, if P_A^k denotes the number of $k \times n$ arrangements corresponding to A, then

$$P_B^{k+1} = 2^{R_{AB}} P_A^k. \tag{1}$$

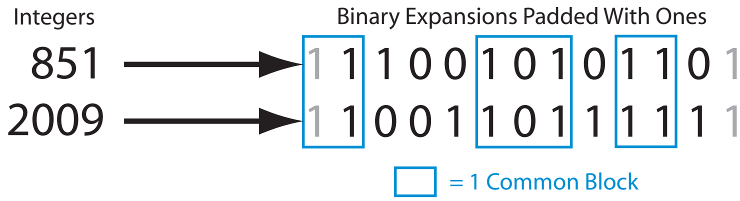
Note that $P_A^1 = 1$ for every A, and so the total number of $m \times n$ arrangements is the sum of the entries comprising S^{m-1} ($m-1$ matrix multiplications), where S is the $2^{n-1} \times 2^{n-1}$ matrix with $S_{ij} = 2^{R_{(i-1)(j-1)}}$. Given the Matlab function `getR.m` for finding R_{AB} (see code on the next page), the total number of $m \times n$ arrangements can be computed as follows.

```
function num=ArrangeNum(m,n)
n=n-1;
R=zeros(2^n);
for r=1:2^n
  for c=r:2^n
    R(r,c)=getR(r-1,c-1,n);
    R(c,r)=R(r,c);
  end
end
num=sum(sum((2.^R)^(m-1)));
```

This number explodes with increase in m and n. There are 322 arrangements on a 3×3 grid, 70,878 arrangements on a 4×4 grid, 84,231,996 arrangements on a 5×5 grid, and 535,236,230,270 arrangements on a 6×6 grid. The solution space of grid based page layouts is vast.

What about R_{AB}?

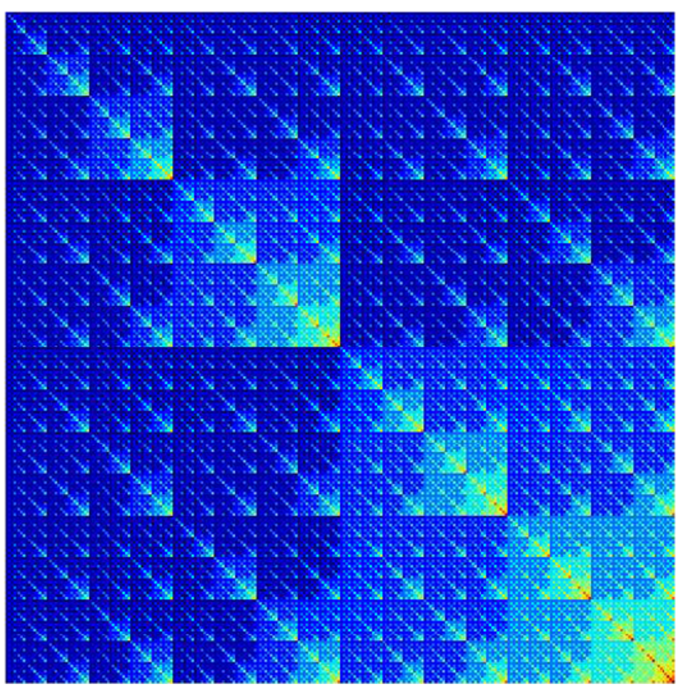
Let $(a_1, a_2, \ldots, a_{n-1})$ be a binary expansion[2] of A, and let $(b_1, b_2, \ldots, b_{n-1})$ be a binary expansion of B. Because binary expansions correspond to spatial arrangements, we care about the extent to which blocks in two expansions overlap. Let $(\tilde{a}_1, \tilde{a}_2, \tilde{a}_3, \ldots, \tilde{a}_n, \tilde{a}_{n+1}) = (1, a_1, a_2, \ldots, a_{n-1}, 1)$, and similarly define the $\tilde{b}_i$'s in terms of the b_i's. A *block* common to A and B is a set of integers from i to $j > i$ for which $\tilde{a}_k = \tilde{b}_k = 0$ for k between i and j, and for which $\tilde{a}_i = \tilde{b}_i = 1$ and $\tilde{a}_j = \tilde{b}_j = 1$.



R_{AB} denotes the number of blocks common to A and B, (for instance with $A = 851$ and $B = 2009$ shown above, $R_{AB} = 3$). R_{AB} can be computed (in Matlab) as follows.

```
function R=getR(A,B,N)
bA=[1 bitget(A,1:N) 1];
bB=[1 bitget(B,1:N) 1];
U=bA|bB;
F=find(bA&bB);
s=zeros(numel(F)-1,1);
for k=1:numel(F)-1
  s(k)=sum(U(F(k):F(k+1)));
end
R=sum(s==2);
```

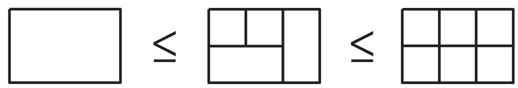
The values R_{ij} have an intriguing fractal character; the following is a scaled image of $[R_{ij}]$ for i and j ranging from 0 to 255.



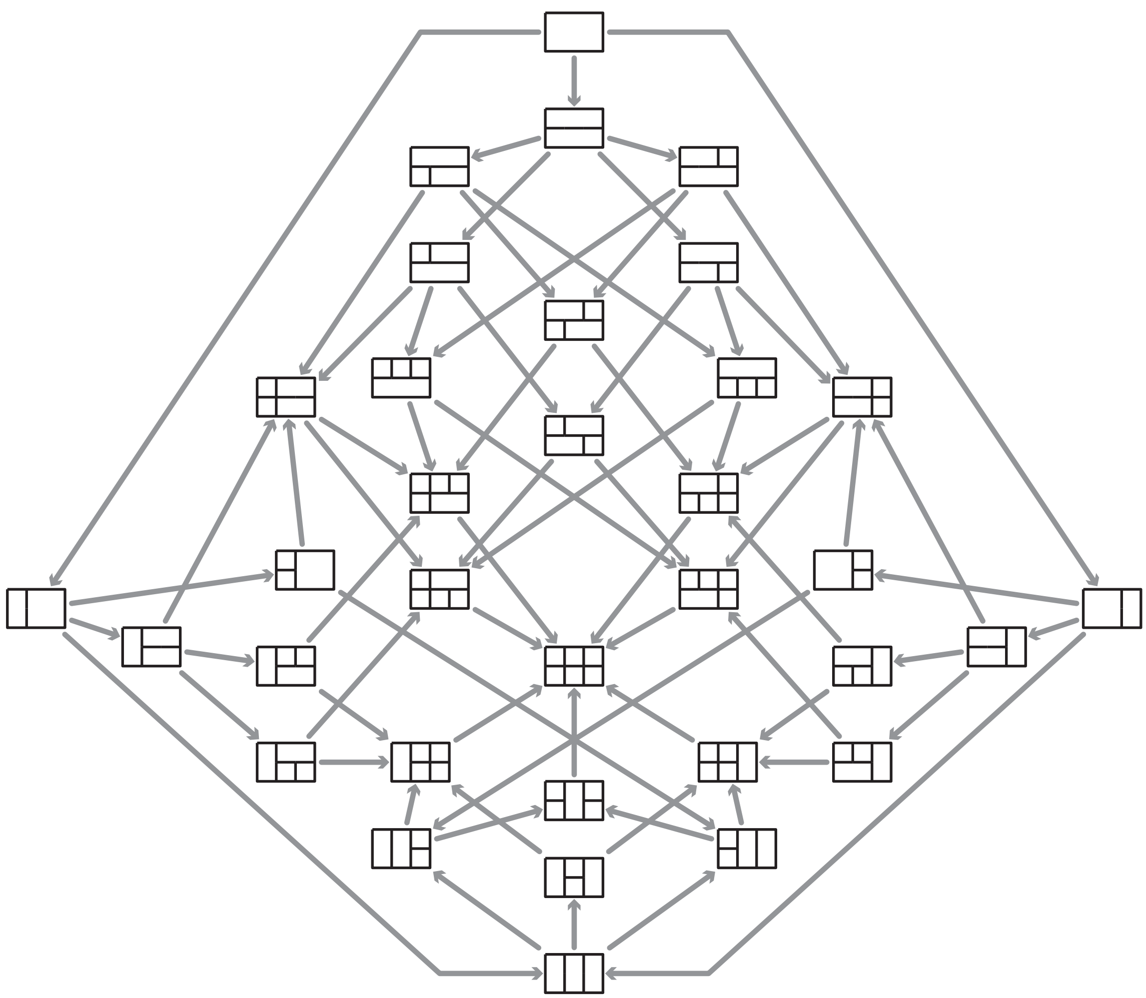
[2]A binary expansion of a non negative integer A is a list $(a_1, a_2, \ldots, a_n)$ of 1's and 0's for which $A = \sum_{k=1}^{n} a_k 2^{k-1}$.

Symmetry
A set of points in $\mathbb{R}^2$ (such as the lines comprising an arrangement of rectangles on a grid) is *symmetric* with respect to the mapping $f : \mathbb{R}^2 \longrightarrow \mathbb{R}^2$ if the set is equal to its image under f. The only mappings that matter for arrangements of rectangles are horizontal and vertical reflections, and 180° rotations, (and 90° rotations when the underlying grid is square). We have considered displaying only one of a set of arrangements related by symmetry (see `http://www.mechanicaldust.com/Documents/3by4.pdf`), however we have abandoned this because it forces the reflection and rotation to be done mentally, (which can lead to error).

A Family Tree
Let $\mathcal{A}$ denote the set of all $m \times n$ arrangements. If every line segment of $a \in \mathcal{A}$ is also present in $b \in \mathcal{A}$, then we say that a is an *ancestor* of b, and that b is a *descendant* of a, and we write $a \leq b$. (The relation $\leq$ is a *partial order* on $\mathcal{A}$.)



Although arrangements may exist which are not related by $\leq$, every arrangement is an ancestor of the arrangement with no inside lines, and a descendent of the arrangement with all inside lines. We call a a *parent* of b (and b a *child* of a) if $a \leq b$ and if there is no arrangement x for which $a \leq x \leq b$. Child-parent relationships link all arrangements together in a network. The network of 2×3 arrangements is shown below, with arrows pointing from parents to children.

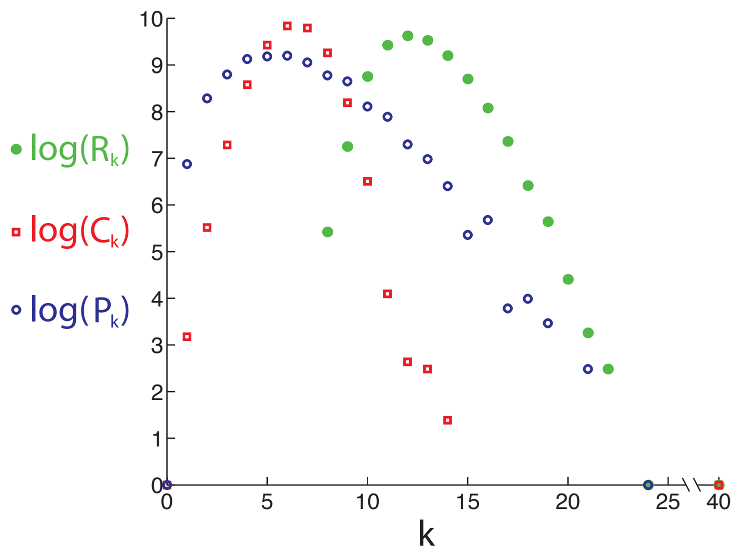


This image of the 2×3 arrangements is a hand crafted mapping of network nodes into the plane of the page. We do our best to keep the nodes evenly spread out, and at the same time to minimize the length of the network edges as well as the number of edge crossings. We also try to arrange the nodes in a way that highlights symmetry relationships.

We lack an automated method of displaying networks in 2D, however this isn't a setback- having such a method wouldn't be helpful because the number of network nodes grows so quickly as the grid gets bigger that any picture of *all* the nodes would be illegible (there are over 80 million nodes for the 5×5 network).

Immediate Network Relatives
The number of immediate network relatives of any arrangement is much lower than the total number of arrangements. In a 4×4 grid for instance, there are 70878 arrangements, and none of these has more than 40 immediate relatives. In the following table and graph, R_k, C_k, and P_k are the number of 4×4 arrangements with k immediate relatives, children, and parents respectively.

k	R_k	C_k	P_k
0	0	1	1
1	0	24	968
2	0	249	3964
3	0	1460	6606
4	0	5309	9211
5	0	12396	9720
6	0	18679	9868
7	0	17880	8556
8	226	10513	6480
9	1410	3608	5702
10	6328	668	3330
11	12356	60	2668
12	15117	14	1477
13	13722	12	1076
14	9907	4	604
15	6002	0	212
16	3221	0	292
17	1574	0	44
18	611	0	54
19	282	0	32
20	82	0	0
21	26	0	12
22	12	0	0
24	1	0	1
40	1	1	0



The number of steps between immediate relatives needed to get from one node to another is also very small.

Navigating Arrangements
We define the *navigability* of a network by the quotient RS/N, where R is the maximum number of immediate node relatives (over all nodes), where S is the maximum, over all pairs of nodes, of the minimum number of network steps between the pair, and where N is the total number of nodes. Our network has high navigability, and so it's easy to get from one node to another by jumping between immediate network relatives. We have posted a Java applet for doing this at `http://www.mechanicaldust.com/applet/index.html`.

References

[1] Karl Gerstner. *Designing Programmes*. 1964.

[2] Le Corbusier (Charles Edouard Jeanneret). *Le Modulor*. 1948.

[3] Josef Muller-Brockman. *Grid Systems*. 1981.

[4] Joshua Smith and Helena Verrill. On Dividing Rectangles Into Rectangles. *Fibonacci Quarterly, (to appear)*. Preprint at `http://www.math.lsu.edu/~verrill/research/rectangles.pdf`.