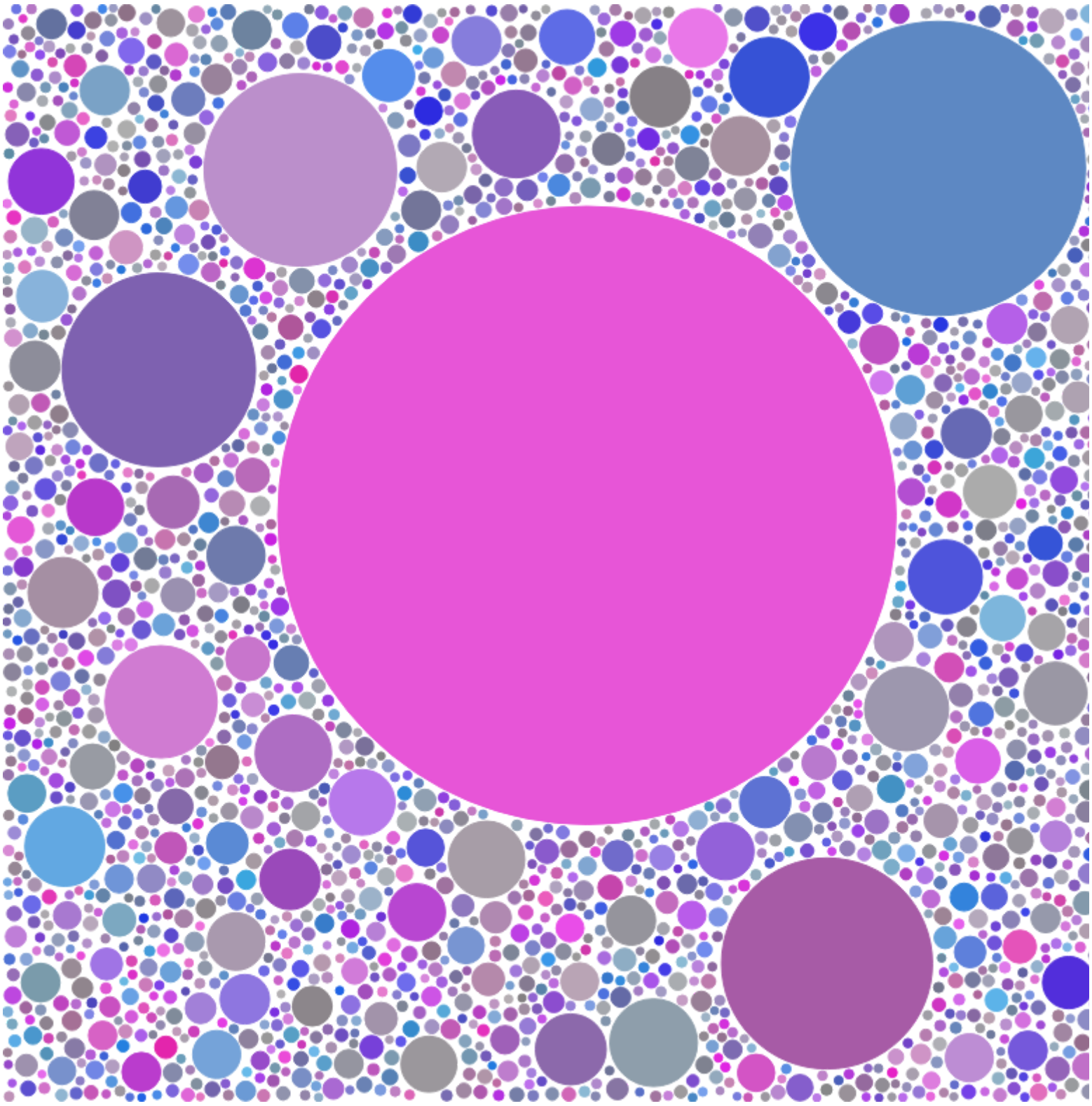
Algorithmic Art

We want to generate probabilistic artwork, efficiently. We are going to use random variables to make a picture filled with non-overlapping circles:



Regenerate

In our art, the circles are different sizes. Specifically, each circle's **radius** is drawn from a Pareto distribution (which is described below). The placement algorithm is greedy: we sample 1000 circle sizes. Sort them by size, largest to smallest. Loop over the circle sizes and place circles one by one.

To place a circle on the canvas, we sample the location of the center of the circle. Both the x and y coordinates are uniformly distributed over the dimensions of the canvas. Once we have selected a prospective location we then check if there would be a collision with a circle that has already been placed. If there is a collision we keep trying new locations until you find one that has no collisions.

The Pareto Distribution

Pareto Random Variable

Notation:	$X \sim \text{Pareto}(a)$
Description:	A long tail distribution. Large values are rare and small values are common.
Parameters:	$a \geq 1$, the shape parameter Note there are other optional params. See wikipedia

Support:	$x \in [0, 1]$
PDF equation:	$f(x) = \frac{1}{x^{a+1}}$
CDF equation:	$F(x) = 1 - \frac{1}{x^a}$

Sampling from a Pareto Distribution

How can we draw samples from a pareto? In python its simple: `stats.pareto.rvs(a)` however in JavaScript, or other languages, it might not be made transparent. We can use "inverse transform sampling". The simple idea is to choose a uniform random variable in the range (0, 1) and then select the value x assignment such that $F(x)$ equals the randomly chosen value.

$$
\begin{aligned}
y &= 1 - (\frac{1}{x})^{\alpha} \\
(\frac{1}{x})^{\alpha} &= 1 - y \\
\frac{1}{x} &= (1 - y)^{\frac{1}{\alpha}} \\
x &= \frac{1}{(1 - y)^{\frac{1}{\alpha}}}
\end{aligned}
$$