Macvlan 网络方案实践

通过实验来验证 Macvlan Bridge 模式的连通性

发表于 April 1, 2019

Mon Apr 1, 2019

1600 Words|Read in about 4 Min|本文总阅读量次

Tags: macvlan (https://www.yangcs.net/tags/macvlan/)

通过上篇文章 (https://www.yangcs.net/posts/netwnetwork-virtualization-macvlan/)的学习，我们已经知道 Macvlan 四种模式的工作原理，其中最常用的就是 Bridge 模式，本文我们将通过实验来验证 Macvlan Bridge 模式的连通性。

Macvlan 是 linux 内核比较新的特性，可以通过以下方法判断当前系统是否支持：

```
$ modprobe macvlan
$ lsmod | grep macvlan
  macvlan                19233  0
```

如果第一个命令报错，或者第二个命令没有返回，则说明当前系统不支持 Macvlan，需要升级系统或者升级内核。

1. 各个 Linux 发行版对 Macvlan 的支持

Macvlan 对 Kernel 版本依赖： `Linux kernel v3.9–3.19 and 4.0+` 。几个重要发行版支持情况：

- ubuntu： >= saucy(13.10)
- RHEL(Red Hat Enterprise Linux): >= 7.0(3.10.0)
- Fedora: >=19(3.9)
- Debian: >=8(3.16)

各个发行版的内核都可以自行手动升级，具体操作可以参考官方提供的文档。

以上版本信息参考了这些资料：

- List of ubuntu versions with corresponding linux kernel version (https://askubuntu.com/questions/517136/list-of-ubuntu-versions-with-corresponding-linux-kernel-version)
- Red Hat Enterprise Linux Release Dates (https://access.redhat.com/articles/3078)

2. 实验环境

后面的测试将会在以下环境进行：

OS	hostname	物理网卡	IP	Gateway
CentOS 7.3	node1	ens160	192.168.179.$^{9}/_{16}$	192.168.1.1
CentOS 7.3	node2	ens160	192.168.179.$^{10}/_{16}$	192.168.1.1

我的本地操作系统为 MacOS，IP 为 `10.8.0.241` ，网关为 `10.8.0.1` 。

3. 连通性测试

下面开始对 Bridge 模式下 Macvlan 的连通性进行测试。

首先在 `node1` 上创建两个 network namespace：

```
# 开启混杂模式
$ ip link set ens160 promisc on

$ ip netns add ns1
$ ip netns add ns2
```

然后创建 Macvlan 接口:

```
$ ip link add link ens160 mac1 type macvlan mode bridge
```

创建的格式为 `ip link add link <PARENT> <NAME> type macvlan mode <MODE>`，其中 `<PARENT>` 是 Macvlan 接口的父接口名称，`<NAME>` 是新建的 Macvlan 接口的名称，这个名字可以任意取，`<MODE>` 是 Macvlan 的模式。

可以查看创建接口的详细信息:

```
$ ip -d link show mac1

13: [email protected]: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN mod
    link/ether 5a:94:85:a6:96:95 brd ff:ff:ff:ff:ff:ff promiscuity 0
    macvlan  mode bridge addrgenmode eui64
```

下面就是把创建的 Macvlan 接口放到 network namespace 中，配置好 IP 地址，然后启用它:

```
$ ip link set mac1 netns ns1
$ ip netns exec ns1 ip addr add 192.168.179.12/16 dev mac1
$ ip netns exec ns1 ip link set dev mac1 up
```

同理可以配置另外一个 Macvlan 接口:

```
$ ip link add link ens160 mac2 type macvlan mode bridge
$ ip link set mac2 netns ns2
$ ip netns exec ns2 ip addr add 192.168.179.13/16 dev mac2
$ ip netns exec ns2 ip link set dev mac2 up
```

可以测试两个 IP 的连通性:

ns1 -> ns2

```
$ ip netns exec ns1 ping -c 3 192.168.179.13

PING 192.168.179.13 (192.168.179.13) 56(84) bytes of data.
64 bytes from 192.168.179.13: icmp_seq=1 ttl=64 time=0.090 ms
64 bytes from 192.168.179.13: icmp_seq=2 ttl=64 time=0.061 ms

--- 192.168.179.13 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1000ms
rtt min/avg/max/mdev = 0.061/0.075/0.090/0.016 ms
```

ns2 -> ns1

```
$ ip netns exec ns2 ping -c 2 192.168.179.12

PING 192.168.179.12 (192.168.179.12) 56(84) bytes of data.
64 bytes from 192.168.179.12: icmp_seq=1 ttl=64 time=0.059 ms
64 bytes from 192.168.179.12: icmp_seq=2 ttl=64 time=0.043 ms

--- 192.168.179.12 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1000ms
rtt min/avg/max/mdev = 0.043/0.051/0.059/0.008 ms
```

ns1 -> 192.168/16

首先测试 ns1 与 node2 的连通性：

```
$ ip netns exec ns1 ping -c 2 192.168.179.10

PING 192.168.179.10 (192.168.179.10) 56(84) bytes of data.
64 bytes from 192.168.179.10: icmp_seq=1 ttl=64 time=0.976 ms
64 bytes from 192.168.179.10: icmp_seq=2 ttl=64 time=0.430 ms

--- 192.168.179.10 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1001ms
rtt min/avg/max/mdev = 0.430/0.703/0.976/0.273 ms
```

下面测试 ns1 与 node2 中 network namespace 的连通性。

先在 node2 中配置一个 Macvlan 接口：

```
[[email protected] ~]# ip link set ens160 promisc on
[[email protected] ~]# ip netns add ns1
[[email protected] ~]# ip link add link ens160 mac1 type macvlan mode bridge
[[email protected] ~]# ip link set mac1 netns ns1
[[email protected] ~]# ip netns exec ns1 ip addr add 192.168.179.14/16 dev mac1
```

```
[[email protected] ~]# ip link set dev mac1 up
```

测试 node1 的 ns1 与 node2 的 ns1 的连通性：

```
[[email protected] ~]# ip netns exec ns1 ping -c 2 192.168.179.14

PING 192.168.179.14 (192.168.179.14) 56(84) bytes of data.
64 bytes from 192.168.179.14: icmp_seq=1 ttl=64 time=0.976 ms
64 bytes from 192.168.179.14: icmp_seq=2 ttl=64 time=0.430 ms

--- 192.168.179.14 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1001ms
rtt min/avg/max/mdev = 0.430/0.703/0.976/0.273 ms
```

10.8%16 -> ns1

```
# 在本地的 MacOS 客户端 ping 192.168.179.12
$ ping 192.168.179.12 -c 2

PING 192.168.179.12 (192.168.179.12): 56 data bytes
Request timeout for icmp_seq 0

--- 192.168.179.12 ping statistics ---
2 packets transmitted, 0 packets received, 100.0% packet loss
```

发现跨三层网段是 ping 不通的。这个问题很好解决，我们刚刚给 ns1 和 ns2 分配 IP 的时候并没有指定默认路由，指定个默认路由问题就迎刃而解了。

```
$ ip netns exec ns1 ip route add default via 192.168.1.1 dev mac1
```

Note

如果你想开发 Macvlan cni 插件，这个地方需要注意一下，每次给 Pod 分配好 IP 以后要添加一条默认路由指向网关，不然无法跨三层通信。

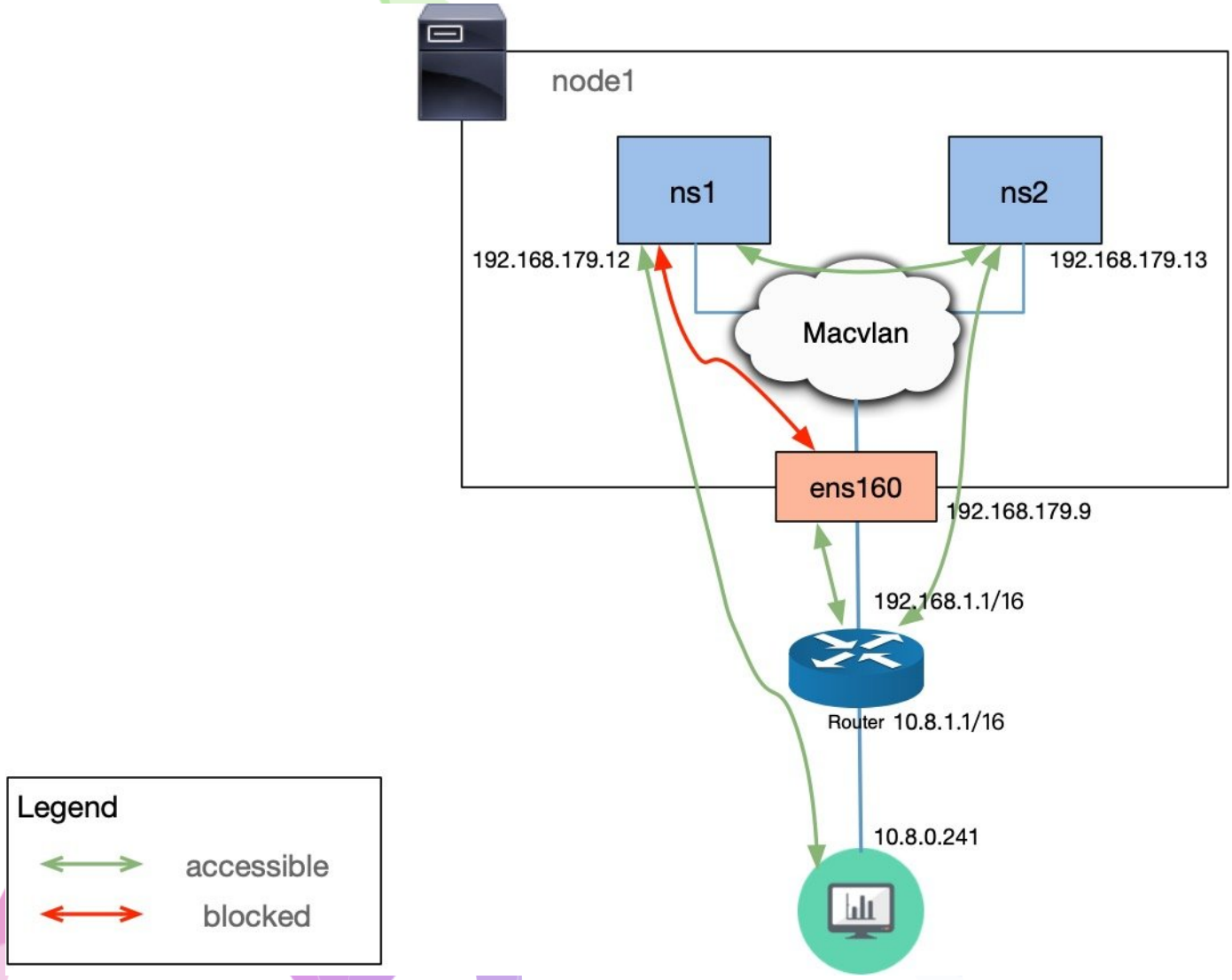
ns1 -> ens160

```
$ ip netns exec ns1 ping -c 2 192.168.179.9

PING 192.168.179.9 (192.168.179.9) 56(84) bytes of data.

--- 192.168.179.9 ping statistics ---
2 packets transmitted, 0 received, 100% packet loss, time 999ms
```

这里就遇到了我在上一篇文章 (https://www.yangcs.net/posts/netwnetwork-virtualization-macvlan/#span-id-inline-toc-1-span-macvlan-简介)开头提到的问题。到目前为止，整个实验的拓扑结构如下：



其实也很好解决，额外创建一个 Macvlan 子接口，并把 ens160 的 IP 分给这个子接口，最后还要修改默认路由。

```
$ ip link add link ens160 mac0 type macvlan mode bridge
# 下面的命令一定要放在一起执行，否则中间会失去连接
$ ip addr del 192.168.179.9/16 dev ens160 && \
  ip addr add 192.168.179.9/16 dev mac0 && \
  ip link set dev mac0 up && \
  ip route flush dev ens160 && \
  ip route flush dev mac0 && \
  ip route add 192.168.0.0/16 dev mac0 metric 0 && \
  ip route add default via 192.168.1.1 dev mac0 &
```

Note

这里一定不能 Down 掉 `ens160`，否则所有的子接口都将无法工作。

现在就能 ping 通了：

```
$ ip netns exec ns1 ping -c 2 192.168.179.9

PING 192.168.179.9 (192.168.179.9) 56(84) bytes of data.
64 bytes from 192.168.179.9: icmp_seq=1 ttl=64 time=0.137 ms
64 bytes from 192.168.179.9: icmp_seq=2 ttl=64 time=0.078 ms

--- 192.168.179.9 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 999ms
rtt min/avg/max/mdev = 0.078/0.107/0.137/0.031 ms
```

--------他日江湖相逢 ☂ 再当杯酒言欢--------

「真诚赞赏，手留余香」

赏

分享到：

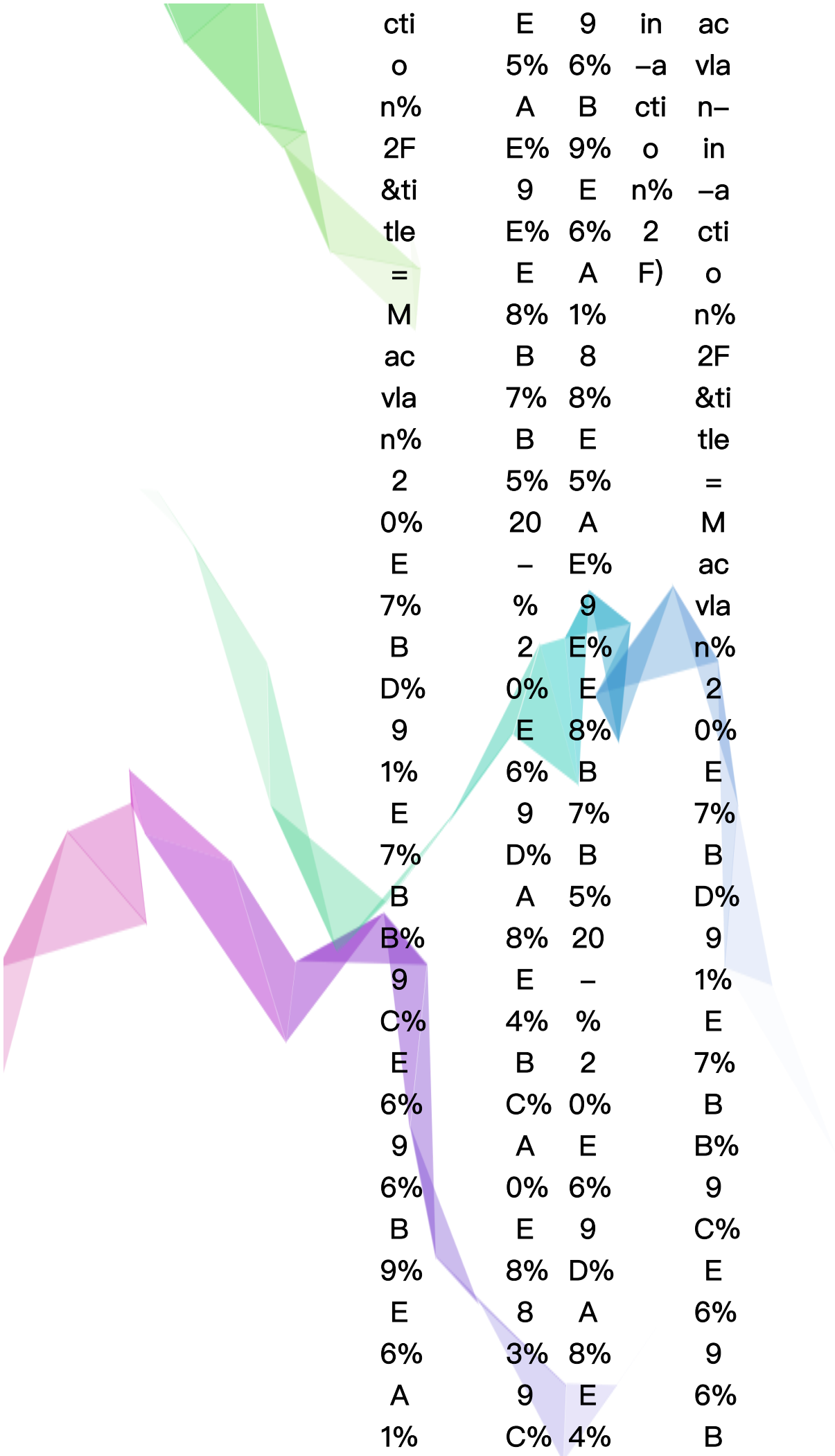
← 前一篇 (HTTPS://WWW.YANGCS.NET/POSTS/NETWNETWORK-VIRTUALIZATION-MACVLAN/)

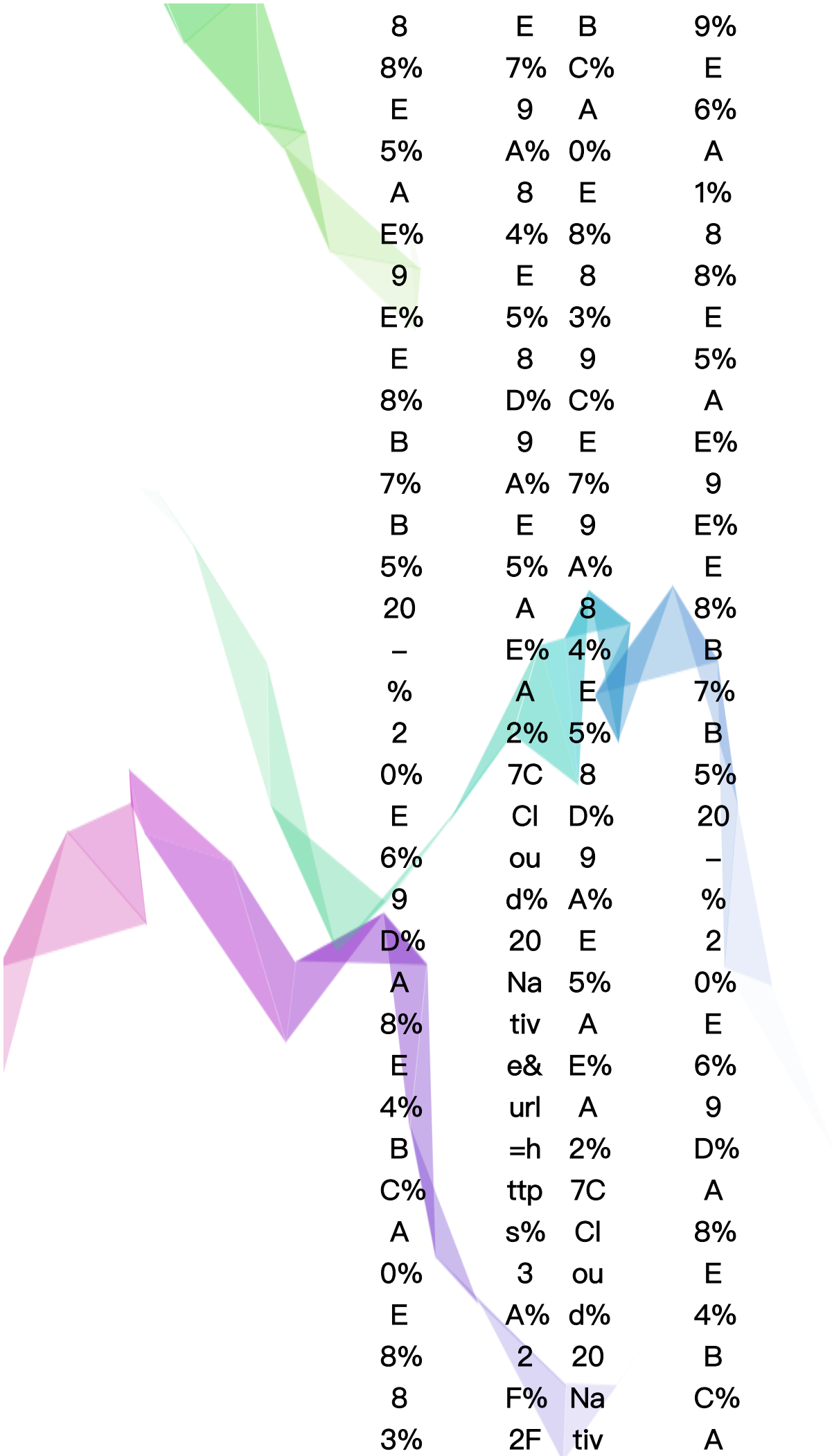
后一篇 → (HTTPS://WWW.YANGCS.NET/POSTS/KUBE-PROXY-SUBTLETIES-DEBUGGING-AN-INTERMITTENT-CONNECTION-RESET/)

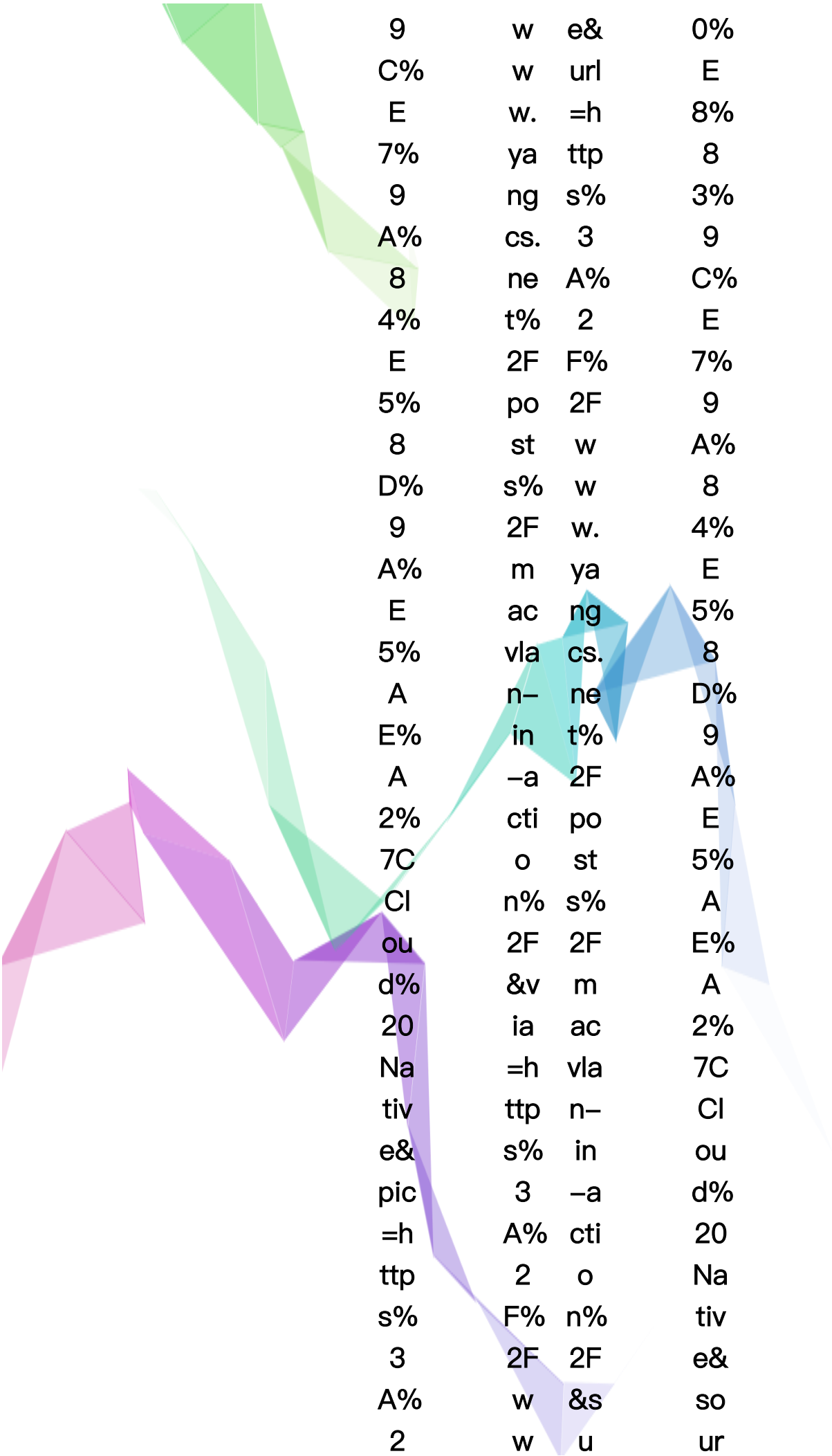
相关文章

- Linux 虚拟网卡技术：Macvlan (/posts/netwnetwork-virtualization-macvlan/)
- 浅析 Kubernetes 控制器的工作原理 (/posts/a-deep-dive-into-kubernetes-controllers/)
- 理解矩阵乘法 (/posts/matrix-multiplication/)

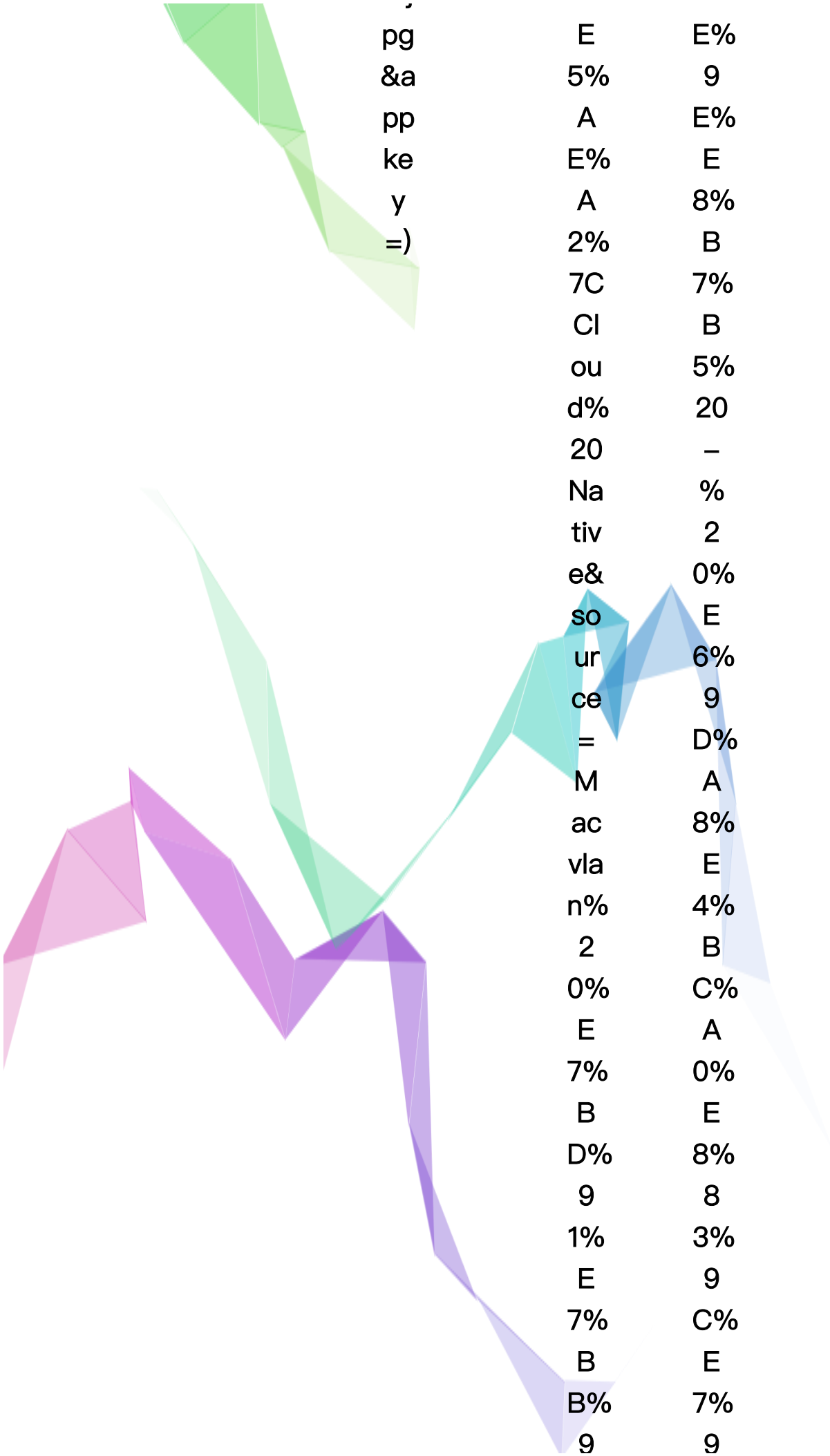
- 使用 CoreDNS 来应对 DNS 污染 (/posts/install-coredns-on-macos/)
- Kubernetes 设计哲学 (/posts/the-mechanics-of-kubernetes/)


cti E 9 in ac
o 5% 6% –a vla
n% A B cti n–
2F E% 9% o in
&ti 9 E n% –a
tle E% 6% 2 cti
= E A F) o
M 8% 1% n%
ac B 8 2F
vla 7% 8% &ti
n% B E tle
2 5% 5% =
0% 20 A M
E – E% ac
7% % 9 vla
B 2 E% n%
D% 0% E 2
9 E 8% 0%
1% 6% B E
E 9 7% 7%
7% D% B B
B A 5% D%
B% 8% 20 9
9 E – 1%
C% 4% % E
E B 2 7%
6% C% 0% B
9 A E B%
6% 0% 6% 9
B E 9 C%
9% 8% D% E
E 8 A 6%
6% 3% 8% 9
A 9 E 6%
1% C% 4% B

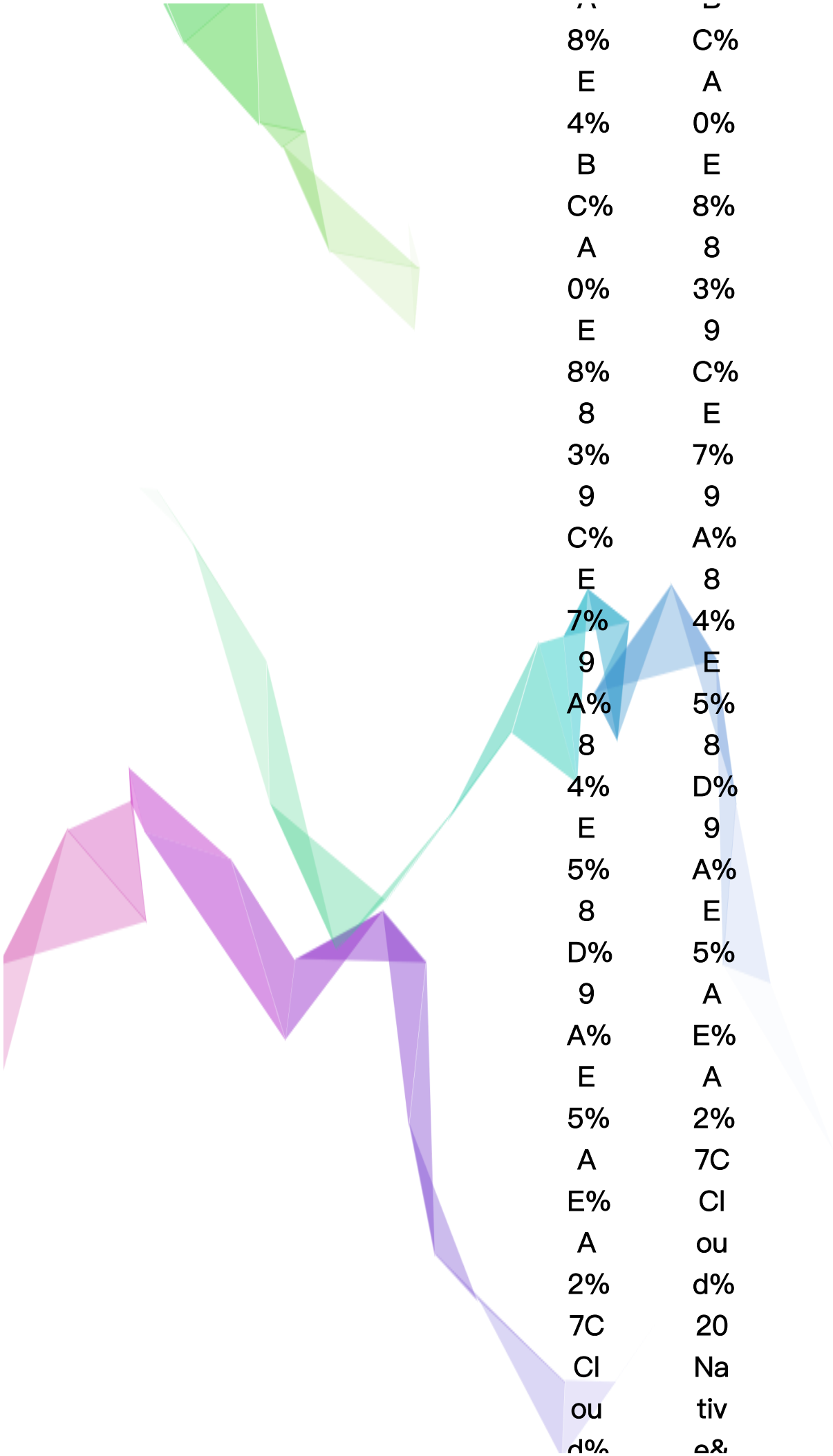

8 E B 9%
8% 7% C% E
E 9 A 6%
5% A% 0% A
A 8 E 1%
E% 4% 8% 8
9 E 8 8%
E% 5% 3% E
E 8 9 5%
8% D% C% A
B 9 E E%
7% A% 7% 9
B E 9 E%
5% 5% A% E
20 A 8 8%
– E% 4% B
% A E 7%
2 2% 5% B
0% 7C 8 5%
E Cl D% 20
6% ou 9 –
9 d% A% %
D% 20 E 2
A Na 5% 0%
8% tiv A E
E e& E% 6%
4% url A 9
B =h 2% D%
C% ttp 7C A
A s% Cl 8%
0% 3 ou E
E A% d% 4%
8% 2 20 B
8 F% Na C%
3% 2F tiv A

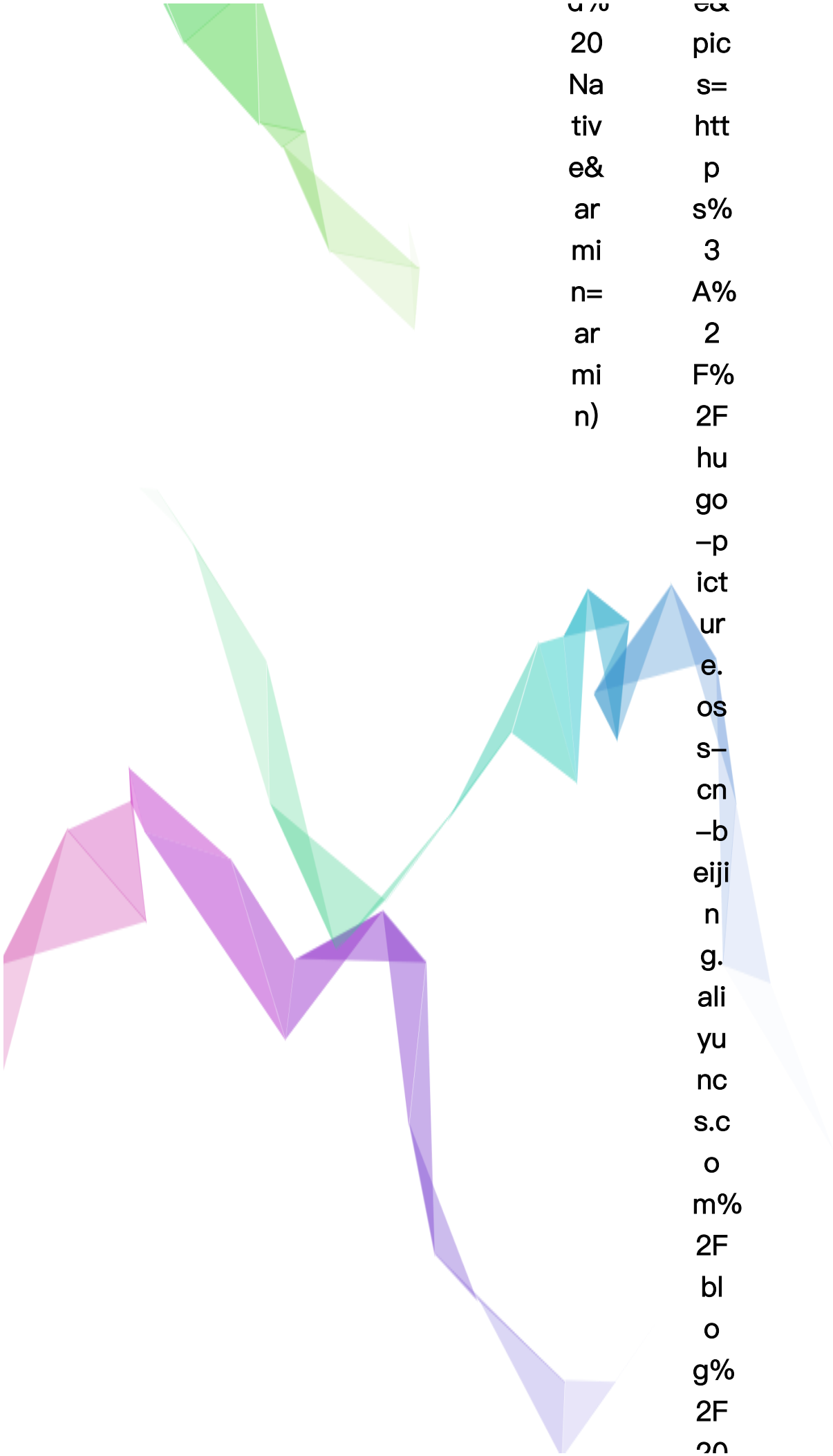

9 w e& 0%
C% w url E
E w. =h 8%
7% ya ttp 8
9 ng s% 3%
A% cs. 3 9
8 ne A% C%
4% t% 2 E
E 2F F% 7%
5% po 2F 9
8 st w A%
D% s% w 8
9 2F w. 4%
A% m ya E
E ac ng 5%
5% vla cs. 8
A n– ne D%
E% in t% 9
A –a 2F A%
2% cti po E
7C o st 5%
Cl n% s% A
ou 2F 2F E%
d% &v m A
20 ia ac 2%
Na =h vla 7C
tiv ttp n– Cl
e& s% in ou
pic 3 –a d%
=h A% cti 20
ttp 2 o Na
s% F% n% tiv
3 2F 2F e&
A% w &s so
2 w u ur


F% w. m ce
2F ya m =
hu ng ar M
go cs. y ac
–p ne =% vla
ict t) E n%
ur 6% 2
e. 9 0%
os D% E
s– A 7%
cn 8% B
–b E D%
eiji 4% 9
n B 1%
g. C% E
ali A 7%
yu 0% B
nc E B%
s.c 8% 9
o 8 C%
m% 3% E
2F 9 6%
bl C% 9
o E 6%
g% 7% B
2F 9 9%
20 A% E
19 8 6%
–0 4% A
4– E 1%
27 5% 8
–1 8 8%
45 D% E
21 9 5%
4.i A% A


pg
&a
pp
ke
y
=)
E
5%
A
E%
A
2%
7C
Cl
ou
d%
20
20
Na
tiv
e&
so
ur
ce
=
M
ac
vla
n%
2
0%
E
7%
B
D%
9
1%
E
7%
B
B%
9
E
9
E%
E
8%
B
7%
B
5%
20
–
%
2
0%
E
6%
9
D%
A
8%
E
4%
B
C%
A
0%
E
8%
8
3%
9
C%
E
7%
9

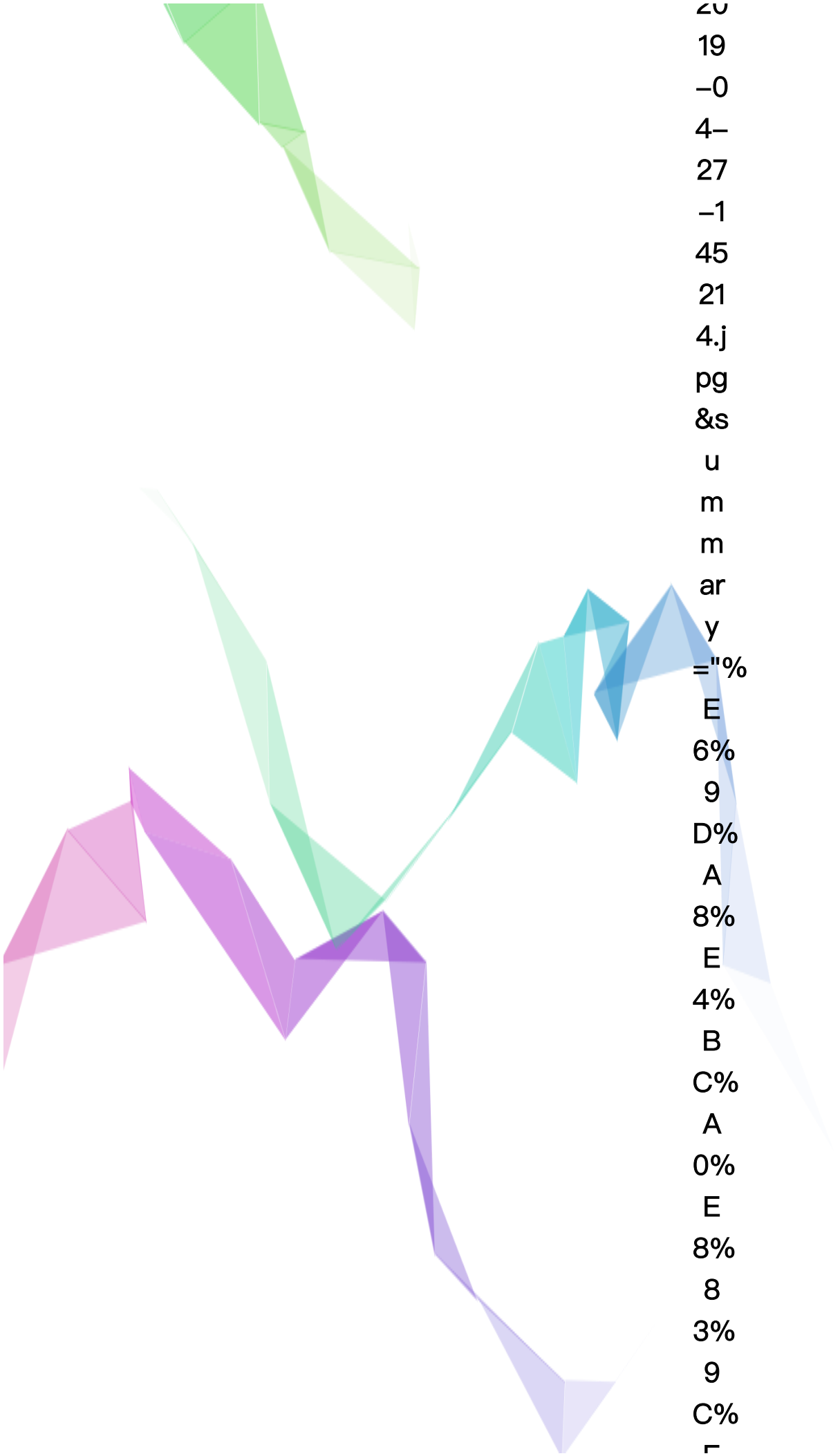

C% A%
E 8
6% 4%
9 E
6% 5%
B 8
9% D%
E 9
6% A%
A E
1% 5%
8 A
8% E%
E A
5% 2%
A 7C
E% Cl
9 ou
E% d%
E 20
8% Na
B tiv
7% e&
B de
5% sc
20 =%
– E
% 6%
2 9
0% D%
E A
6% 8%
9 E
D% 4%
A B


8% C%
E A
4% 0%
B E
C% 8%
A 8
0% 3%
E 9
8% C%
8 E
3% 7%
9 9
C% A%
E 8
7% 4%
9 E
A% 5%
8 8
4% D%
E 9
5% A%
8 E
D% 5%
9 A
A% E%
E A
5% 2%
A 7C
E% Cl
A ou
2% d%
7C 20
Cl Na
ou tiv
d% e&



d%
20
Na
tiv
e&
ar
mi
n=
ar
mi
n)

eq
pic
s=
htt
p
s%
3
A%
2
F%
2F
hu
go
–p
ict
ur
e.
os
s–
cn
–b
eiji
n
g.
ali
yu
nc
s.c
o
m%
2F
bl
o
g%
2F
20


20
19
-0
4-
27
-1
45
21
4.j
pg
&s
u
m
m
ar
y
="%
E
6%
9
D%
A
8%
E
4%
B
C%
A
0%
E
8%
8
3%
9
C%

E
7%
9
A%
8
4%
E
5%
8
D%
9
A%
E
5%
A
E%
A
2%
7C
Cl
ou
d%
20
Na
tiv
e")