Resolve the inconsistency of Illumina identifiers through nuID

Pan Du[‡,*], Jared Flatow[‡,†], Warren A. Kibbe[‡,‡], Simon Lin[‡,§]

September 18, 2008

[‡]Robert H. Lurie Comprehensive Cancer Center
Northwestern University, Chicago, IL, 60611, USA

Contents

1 Illumina Identifiers and BeadStudio output files **1**

2 nuID (nucleotide universal IDentifier) **2**
2.1 Examples of nuID 3

3 Illumina microarray annotation packages **3**
3.1 Transfer Illumina identifier annotated data into nuID annotated 4

4 Online service of nuID conversion and updated annotation information **4**

5 References **5**

1 Illumina Identifiers and BeadStudio output files

Illumina uses two types of identifiers: Illumina gene identifiers and Illumina Probe identifiers. As their names suggest, Illumina gene identifiers are designed for genes while Illumina probe identifiers are designed for probes. The problem of the gene identifier is that it can correspond to several different probes, which are supposed to match the same gene. In this case, it basically averages the measurements of these probes. This will cause big problem when these probes for the same gene have different measurement values. This happens often in real situations because of the binding affinity difference or alternative splicing, the probes corresponding the the sample gene identifier may have quite different expression levels and patterns. If we use the gene identifier to identify the measurements, then we cannot differentiate the difference between these probes. Another problem of using gene identifiers is that the mapping between gene

[*]dupan@northwestern.edu
[†]jflatow@northwestern.edu
[‡]wakibbe@northwestern.edu
[§]s-lin2@northwestern.edu

identifiers and probes could be changed with our better understanding of the gene. Therefore, we recommend to use probe identifiers.

Before further discussion, let's describe more details of Illumina BeadStudio output files. BeadStudio usually will export a list of files, which include "Control Gene Profile.txt", "Group Probe Profile.txt", "Samples Table.txt", "Control Probe Profile.txt", "Sample Gene Profile.txt", "Group Gene Profile.txt", "Sample Probe Profile.txt". Among these files, the files with their name including "Probe" use Illumina probe identifiers, which are supposed to be unique for each probe. The files with their names including "Gene" use Illumina gene identifiers. As the probe identifiers were designed for each probe, we recommend to use "Sample Probe Profile.txt" or "Group Probe Profile.txt" for the data analysis.

One problem of Illumina identifiers (both Illumina gene identifiers and Illumina probe identifiers) is that they are not stable and consistent between versions. For example, the early version of BeadStudio output files use a numeric number as probe identifier, later on it uses the new version of probe identifiers named as `"ILMN_0000"` ("0000" represents a numeric number). Also, the early version of BeadStudio output files use TargetID as gene identifier, later on gene symbols are directly used as the gene identifiers. The Illumina probe identifiers also change over time. Moreover, the identifiers are not unique. For instance, the same 50mer sequence has two different TargetIDs (early version of gene identifiers) used by Illumina: `"GI_21070949-S"` in the `Mouse_Ref-8_V1` chip and `"scl022190.1_154-S"` in the `Mouse-6_V1` chip. This causes difficulties when combining clinical microarray data collected over time using different versions of the chips.

In order to get unique mapping between microarray measurements and probes, using ProbeID is preferred. However, the ProbeID of Illumina is not stable. It is changing between different versions, even between different batches of Illumina microarrays. To solve these problems, we designed a nucleotide universal identifier (nuID), which encodes the 50mer oligonucleotide sequence and contains error checking and self-identification code. By using nuID, all the problems mentioned above can be easily solved. For details, please read [1].

2 nuID (nucleotide universal IDentifier)

Oligonucleotide probes that are sequence identical may have different identifiers between manufacturers and even between different versions of the same company's microarray; and sometimes the same identifier is reused and represents a completely different oligonucleotide, resulting in ambiguity and potentially mis-identification of the genes hybridizing to that probe.

We have devised a unique, non-degenerate encoding scheme that can be used as a universal representation to identify an oligonucleotide across manufacturers. We have named the encoded representation 'nuID' , for nucleotide universal identifier. Inspired by the fact that the raw sequence of the oligonucleotide is the true definition of identity for a probe, the encoding algorithm uniquely and non-degenerately transforms the sequence itself into a compact identifier (a lossless compression). In addition, we added a redundancy check (checksum) to validate the integrity of the identifier. These two steps, encoding plus checksum, result in an nuID, which is a unique, non-degenerate, permanent, robust and efficient representation of the probe sequence. For commercial applications that require

the sequence identity to be confidential, we have an encryption schema for nuID. We demonstrate the utility of nuIDs for the annotation of Illumina microarrays, and we believe it has universal applicability as a source-independent naming convention for oligomers.

The nuID schema has three significant advantages over using the oligo sequence directly as an identifier: first it is more compact due to the base-64 encoding; second, it has a built-in error detection and self-identification; and third, it can be encrypted in cases where the sequences are preferred not to be disclosed.

2.1 Examples of nuID

```
> library(lumi)

This is mgcv 1.4-1

> ## provide an arbitrary nucleotide sequence as an example
> seq <- 'ACGTAAATTTCAGTTTAAAACCCCCCG'
> ## create a nuID for it
> id <- seq2id(seq)
> print(id)

[1] "YGwPOvwBVW"
```

The original nucleotide sequence can be easily recovered by `id2seq`

```
> id2seq(id)

[1] "ACGTAAATTTCAGTTTAAAACCCCCCG"
```

The nuID is self-identifiable. `is.nuID` can check the sequence is nuID or not. A real nuID

```
> is.nuID(id)

[1] TRUE
```

An random sequence

```
> is.nuID('adfqeqe')

[1] FALSE
```

3 Illumina microarray annotation packages

As the identifier inconsistency between different versions or even different releases of Illumina chips, it makes create annotation packages in the traditional way difficult. In traditional way, we have to create individual annotation packages for different identifiers and different versions and releases of chips. Since that will produce lots of packages, it will make the maintenance difficult and users hard to decide which package to use. Because all the Illumina microarrays use 50-mers, by using the nuID universal identifier, we are able to build one

annotation database for different versions and releases of the human (or other species) chips. Moreover, the nuID can be directly converted to the probe sequence, and used to get the most updated refSeq matches and annotations. The recent transition of Bioconductor annotation packages to use SQLite databases made the package size is no longer a concern.

The latest version of Illumina annotation packages indexed by nuID are based on SQLite databases. They were build by using functions in AnnotaionDbi package. These packages for different Illumina expression chips can be downloaded from Bioconductor. There are three packages: lumiHumanAll.db, lumiMouseAll.db and lumiRatAll.db for three species Human, Mouse and Rat respectively. These packages include all the previously released Illumina expression chips. Basically, we converted the probe sequence as nuID and then pool them together. The mapping between nuID and genebank, refseq, unigene IDs were based on the Illumina Manifest files of the chips. If there are duplicated nuIDs, then the latest version of mapping were used. The usage of these annotation files is exactly the same as other Bioconductor annotation packages, like Affymetrix. The previous versions of packages: lumiHumanV1, lumiHumanV2, lumiMouseV1 and lumiRatV1 will be discontinued.

Need to mention, currently there are two sets of Illumina annotation packages in Bioconductor. The Illumina annotation packages mentioned here are named as "lumixxxx", e.g. "lumiHumanAll.db" and are maintained by us. There are another set of packages, named as "illuminaxxxx". These packages are indexed based on Illumina IDs. They can also be used together with *lumi* package.

3.1 Transfer Illumina identifier annotated data into nuID annotated

As the latest BeadStudio output files will include probe sequence information, by default, lumiR function will automatically convert them as nuID. As a result, the LumiBatch object will be nuID annotated if the BeadStudio output file includes probe sequence information. If the data file do not include the probe sequence information, we provide a online service to convert the Illumina IDs as nuIDs. We will describe more details of online service in the next section. For old versions of Illumina chips (version 1 and 2 of Human chips, version 1 and 1.1 of Mouse chips and version 1 of Rat chip), you can use the previously released Illumina annotation packages (lumiHumanV1, lumiHumanV2, lumiMouseV1 and lumiRatV1) to convert between Illumina IDs and nuIDs (using functions TargetID2nuID, ProbeID2nuID). Function addNuId2lumi can add nuIDs to the LumiBatch object based on probe sequence information or user provided annotation file (or manifest file) for the chip.

4 Online service of nuID conversion and updated annotation information

This part is still under developing. Please check the developing version for the latest updates.

5 References

Du, P., Kibbe, W.A. and Lin, S.M., "nuID: A universal naming schema of oligonucleotides for Illumina, Affymetrix, and other microarrays", Biology Direct 2007, 2:16 (31May2007).