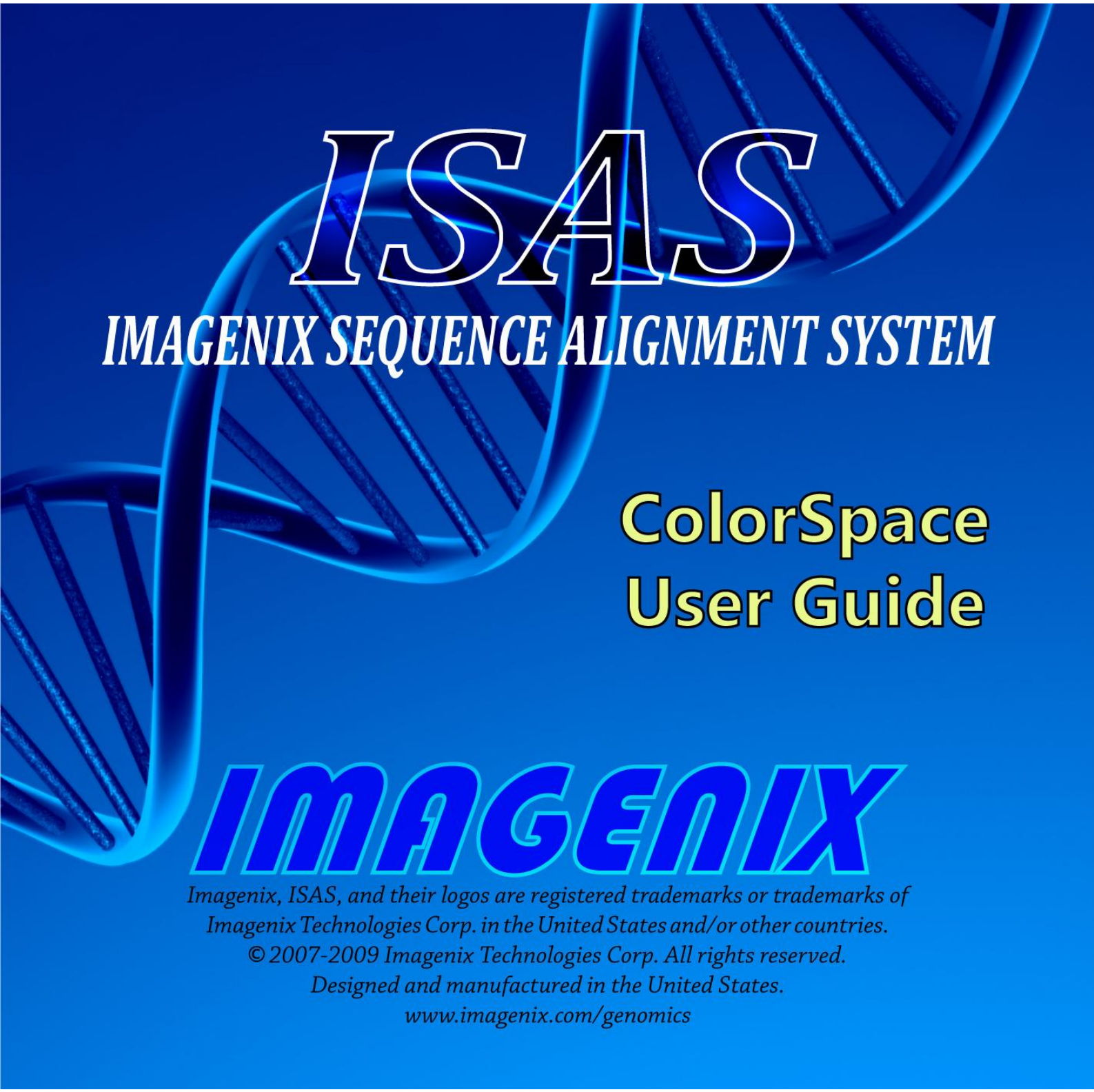

Table of Contents Page

I. Hardware Requirements

If you purchased ISAS as a software package to run on your own hardware, you should pay attention to this section. If you purchased a turn-key system (ISAS software pre-installed on an Imagenix computer) then you can skip this section.

1. CPU

ISAS is designed to run on x86_64 CPUs with SSE3 capabilities. This includes Intel CPUs released since 2004 and AMD CPUs since 2005. You can confirm your processor's capabilities with your computer vendor. Imagenix can also supply you a simple program to interrogate your CPU for its capabilities. ISAS also reports the CPU capabilities it detects.

The installation DVD includes two executables. "ISAScolorsNewCPU" will run only on the newest CPUs (Intel Nehalem family or AMD Shanghai/Istanbul family). While "ISAScolorsOldCPU" is for the older generation of CPUs. A computer with the older CPUs will refuse to run the "NewCPU" version, while the latest generation CPUs can run either versions of software, but will be usually be anywhere from 5 to 30 percent faster with the "NewCPU" version.

While ISAS will run on a system with merely a single CPU core, it is much more cost effective (compared with clusters of computers) to use a system with multiple cores. ISAS makes efficient use of multiple CPUs and multiple cores. The constraint on the number of cores is the ability of the computer's memory controller and data bus to efficiently share the computer's RAM between all the CPU cores. At the time of writing of this guide, 8 cores (e.g. 2 CPUs with 4 cores each) are the most popular platform for ISAS (as well as most high performance applications), achieving almost 7 times the throughput of a single core system. Recently the cost and efficiency of hardware designs of 16 or more cores per computer have become much more attractive than one year ago, and we expect these more powerful systems to become more popular in the very near future. ISAS software is already designed to take advantage of this hardware architecture.

2. RAM

For small organisms, the memory requirement is usually trivial. But for larger reference genomes, we must pay attention to the memory requirements.

If your computer doesn't have enough RAM, and needs to use "virtual RAM" (also called "swapping", where hard disk is used to temporarily replace RAM) then your computer can come to a standstill. The minimum amount of RAM needed to run ISAS depends on the size of the reference genome. While a small organism like E.Coli with a 5 Million base (5MB) genome can easily be aligned on a laptop with only 2GB Ram, the human genome, with 3 billion bases (3GB) requires a minimum of 16GB to run. The minimum requirement is 5bytes of RAM for each base of reference genome, plus roughly 1GB overhead. In addition to the minimum requirement to run, there is an additional requirement on RAM to run efficiently. This additional RAM requirement depends on the settings used: the sequence length and the limit on the number of matches for each sequence. The table below shows how many sequences can be efficiently aligned at a time with various typical settings and total system RAM size, using the 3GB human genome. If you run more

sequences than your RAM permits, ISAS will chop the data into manageable chunks, and run as many iterations as needed. Each iteration incurs a repeat of the overhead of loading the database into memory, which can be from 5 to 20 minutes (depending on your hard disk speed) per iteration. The total run time will be "stepwise linear" with the number of sequences: purely linear alignment run time (we call "R"), plus one fixed hard disk loading time (we call "L") for each iteration.

Total RAM Size	Limit=2			Limit=10		
	Length 25	Length 35	Length 50	Length 25	Length 35	Length 50
32GB	800 Million	604 Million	604 Million	264 Million	237 Million	237 Million
24GB	449 Million	334 Million	334 Million	146 Million	131 Million	131 Million
16GB	86 Million	64 Million	64 Million	28 Million	25 Million	25 Million

No. of Sequences Aligned During Each Iteration for Human Reference.

"Limit=2" means that the maximum number of matches searched for is 2, so for each sequence, alignment will return one of three possible outcomes: the sequence is not found (0 matches), the sequence is found in exactly one genomic position (1), or the sequence is found but in more than one genomic position (2+). "Limit=10" means that the maximum number of matches searched for is 10. For each sequence, there can be 11 outcomes: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, or 10+.

3. Hard Disk

ISAS requires approx. 75GB (or 140GB for "valid adjacent" color space alignment) of disk space, plus the space for the input and output files. During a run, tens of GB of hard disk files are read or written, so a slow hard disk system, whether local or remote (NAS), can waste 5 to 20 minutes per run (or per iteration, if the number of sequences exceeds the iteration size, based on your RAM, see 2 above). It is easy to calculate how much time is "wasted" due to a slow hard disk: for a regular run on 100 Million sequences, we assume roughly 100GB is read from the disk. On the Imagenix GenomeCruncher computer, at the time of writing this document, the sustained disk read rate is 400MB/sec, so this is done in 4 minutes. If your computer has a sustained hard disk read rate of 80MB/sec (typical for off-the-shelf servers), the same task will take 21 minutes, resulting in a total of 17 minutes extra run time. When running in "valid adjacent" mode, the numbers are almost double these.

4. DVD Drive

ISAS software is usually supplied on a DVD-R disk, together with (as a convenience to our customers) the "hg19" public human genome reference. If your system has a DVD-R drive (which is common on modern computers) then you can easily load everything in a few minutes. Note: some computers have a DVD+R drive which cannot read DVD-R disks. Also, we found many Linux systems only allow "root" (superuser) to use the DVD drive. If your computer does not have a compatible DVD reader, but is connected through a network to another computer with a compatible DVD reader, you can read the DVD

contents on the other computer in a few minutes, and copy them over your network in a few more minutes. In the case where you cannot read the DVD in any way, we can supply the software on a CD-R disk, or on a floppy disk, or on a USB flash drive, or by downloading from our web site. You can then download the reference genome over the internet (this can take an hour or more, depending on your connection bandwidth). Note that Linux has the tendency to assign “read only” attributes to files and directories which are copied from a DVD. Make sure to add write (“w”) privileges to the all the directories and files copied, and the execute (“x”) privilege to the executable files.

5. USB Port

In order to run, ISAS relies on a small hardware device, called a “dongle”, which needs to be plugged into the USB port on your computer. This is hardly a “hardware requirement” as virtually all computers made in the last ten years, from simple laptops to advanced servers, have a USB (Universal Serial Bus) port, and it is rare not to have many such ports on a modern computer.

II. Software Requirements

ISAS is available in Linux or Windows versions. In either case, a 64-bit operating system is required. The user must be able to use the USB port at a low level. Linux systems are usually configured in such a way that only “root” (system administrator) or “privileged” users (with system administrator powers) can access the USB port at a low level. Since ISAS uses a “dongle”, a device connected to the USB port, the Linux system should either be set up to allow every user to use the USB port, or the user should have “root” privileges. In addition, many Linux installations were done with “swappiness=60” which is possibly appropriate for desktops, but deleterious for server performance. The value of swapinnes should be set to zero. For both of these Linux configuration problems, see the “Linux Settings” section of “Installing ISAS” below.

III. Installing ISAS

If you purchased a turn-key system (ISAS software pre-installed on an Imagenix computer) then you can skip the “New Installation” section. You are already able to run alignments on the human reference genome. To add or switch to different species, read the section that follows it: “Adding or Changing Reference Genomes“.

1. New Installation

A. Linux Setting - Swappiness

Linux has a parameter called “swappiness”, which needs to be set to zero, otherwise the system will swap memory needlessly, slowing down ISAS. Many Linux installation disks come with a non-zero default (e.g. 60) which may be appropriate for desktops, but will reduce performance of servers, and can devastate the performance of ISAS (especially on large runs). You can set swappiness to zero each time you run ISAS by typing (you need to be “root” or “superuser” to do this):

```
echo 0 > /proc/sys/vm/swappiness
```

You can always check the current swappiness value by typing (no need to be “root”)

cat /proc/sys/vm/swappiness

A better alternative is to permanently set it to zero. As “root”, or “superuser”, edit the file

/etc/sysctl.conf

In the above file, we need to add the following line:

vm.swappiness=0

If, for example, there is already a line such as

vm.swappiness=60

then instead of adding a new line, just change the “60” (in the example above) to “0”. A re-boot is required to load the new swappiness value from this file, after which it is still prudent to check (“cat /proc/sys/vm/swappiness”) that the value is truly “0”.

B. Linux Setting - Regular Users Accessing USB Port

When installing a new ISAS software on your own computer (as opposed to running a turn-key system preinstalled ISAS software on an Imagenix computer), we recommend that you first verify that you can successfully run as root, and only afterwards, if required, add the capability for regular (non root) users to access the USB port at a low level, enabling them to run ISAS. When you reach that stage, return to this section. Some administrators find it easier to use the “sudo” command to allow non-root users to use ISAS than to add privileges for non-root users to access the USB ports.

While different Linux distributions can have different mechanisms for this, below we describe the simple way to grant non-root users permission to access the USB device, which has been tested on Redhat Enterprise 4+, CentOS 4+, Fedora 5+, and SUSE 10+.

As root (or super user), go to the directory

/etc/udev/rules.d

There can be several files there. These define the rules (e.g. permissions) for using devices plugged in to the USB ports. The rules from the different files are applied in alphabetical order of the file names. Create the file

ISAS.rules

With the following content

SYSFS{idVendor}=="07f2", SYSFS{idProduct}=="0001", MODE="0666"

A re-boot may be required to take effect. Now non-root users can access the USB device.

C. Begin Installation

ISAS came with a DVD (disk) and a dongle (small USB device). Insert the disc into your CD/DVD reader, and insert the dongle into one of your USB ports. Make sure the dongle is fully inserted, and the connection is stable.

Create a directory on your hard disk for ISAS, for example “/home/ISAS-colors”. We will use this example in our description, but you can use any other valid directory name. Copy

the contents of the ISAS DVD to “/home/ISAS-colors”, preserving the sub-directory file structure. The directory “/home/ISAS-colors /hg19” will be used to build the reference database, and its subdirectory “/home/ISAS-colors /hg19/reference” will hold the reference genome.

Important Note: After copying from the DVD to your hard disk, some Linux systems may automatically set the permissions of files and/or directories as “read only” (i.e. permission “r” as opposed to “rw”). This can prevent ISAS from functioning properly. Make sure you have write privilages (“rw”) to the ISAS directory and its two subdirectories (“hg19” and “hg19/reference”), as well as all the files - especially the two files “ReferenceDirectory.txt” and “hg19/settings-colors.txt”. You should also have “execute” privilege (“rx”) to the executable file.

For your convenience, the ISAS DVD already includes a compressed version of the public ”hg19” human genome reference, so if copied correctly, the directory “/home/ISAS-colors /hg19/reference” should now contain 75 files: 3 for each of the 24 chromosomes (X was renamed 23 and Y was renamed 24), plus 3 for mitochondrial DNA (renamed 25). The executable program file will be called “ISAScolorsXX” where XX denotes the version, for example “ISAScolorsNewCPU” for latest generation computers, or “ISAScolorsOldCPU” for computers based on previous generation CPUs. You can always try to run ISAScolorsNewCPU and it will tell you if your computer is not compatible, and then you know that you should be using the other version. Depending on your DVD read rate and hard disk write rate, copying should take a few minutes. We suggest first completing the installation for the human reference genome, and getting familiar with ISAS before changing or adding different species/reference.

Change directory to the ISAS directory (type “cd /home/ISAS-colors” in our example) and run ISAS (type “./ISAScolorsNewCPU” or whatever the name of your executable is). Because this is the first time, ISAS will emit error statements, complaining about missing “bin” files, which is normal (you will use ISAS to create these special BIN files later). However, if an error statement is “not installed correctly”, then the USB dongle is not present, or your computer is unable to access the USB port (see section II above). If the error message is “reference not loaded properly”, then you haven’t copied all the files from the DVD into the “/home/ISAS-colors /hg19/reference” directory.

We will now create the missing bin files. The default mode is “regular” as opposed to “valid adjacent”. To be able to perform alignment on the human reference genome, without “valid adjacent”, type “MAKEBIN”, and then press the ENTER key. ISAS will ask “Are you sure ?”, at which point you should type “YES” and press the ENTER key once more. ISAS will now prepare the optimized database for the current reference. This process takes about 30 minutes, and consumes about 75GB of disk space. If you wish to do “valid adjacent” alignments, then first type “MODE=2VA” (followed by ENTER), then give the MAKEBIN command. It takes approximately an hour and 140GB to prepare the database this time. You can sequentially prepare both databases if you plan to perform both types of alignments in the future. In general, after “MODE=2”, a MAKEBIN will create the regular database, and after “MODE=2VA”, a MAKEBIN will create the VA database. Make sure you have the available disk space, or the process will not be able to complete, and you’ll get an error like “error writing…”. When finished, exit ISAS by typing “EXIT” or “QUIT” (followed by the ENTER key). You’re ready to go to section IV.

2. Adding or Changing Reference Genomes

When ISAS is run, it expects the reference and its database to be in a specially structured database subdirectory. It will also create a settings file in the current database subdirectory, which stores your desired settings for using this reference. You can have many different reference genomes to work with, as long as each reference resides in its own directory, and is created and structured correctly. For example, to add "hg18" reference genome (in addition to the "hg19" reference):

a. Create a directory in the location where the ISAS executable resides called "hg18". Create a subdirectory under "hg18" which is called "reference". If we use the example, where ISAS was installed in the directory "/home/ISAS-colors", then the new database directory will be "/home/ISAS-colors/hg18" and its subdirectory will be "/home/ISAS-colors/hg18/reference".

b. Put the "fa" files of all the chromosomes in the "/home/ISAS-colors/hg18/reference" directory. The names should be "chr1.fa", "chr2.fa", etc. Only numbers are allowed, so, in our example, the files "chrX.fa", "chrY.fa", and "ChrM.fa" (if the mitochondrial DNA is also desired) should be renamed to "chr23.fa", "chr24.fa", and "chr25.fa" respectively. For E.Coli, for example, the entire genome should be named "chr1.fa".

c. Run ISAS (if not currently running). Type:
DATABASE=hg18

d. This sets the current reference database to hg18. Send the command by ending with the ENTER key.

e. Set the number of chromosomes by using the "CHR" command. For example, for human with mitochondrial DNA, type "CHR=1,25", without mitochondrial DNA type "CHR=1,24". For E.Coli type "CHR=1,1". The command is sent when you type the ENTER key at the end.

f. Type "FASTA" (followed by ENTER). ISAS will process the "fa" files. This should take a few minutes.

g. Use the MAKEBIN command for either standard, or valid adjacent, or both, as described in the above section III 1. This can take 10 minutes to an hour, depending on your computer.

Note: to switch back to hg19, you can type
DATABASE=hg19
To return to hg18 use the command
DATABASE=hg18

IV. Running ISAS

We recommend that the first time you run ISAS on *your own computer* (i.e. not a turn-key Imagenix system), you do so as "root". This is to avoid a common problem with Linux systems, where normal users are blocked from using the USB port. Only after you are satisfied that the installation is successful, and can run ISAS as root with no difficulties, we suggest that you try to make it useable by non-root users (if required). This is described in section III 1 b on page 6.

ISAS will make use of all of the RAM on your system. It is important not to run any other programs, and to disable unnecessary memory swapping. If you didn't permanently set the "swappiness" value to zero on your system (this is highly recommended), then you will need to remember to set it to zero just before running ISAS every time (see section III 1 on page 5 for instructions). Many Linux installations have a default value of 60 for swappiness, which causes the system to try to "predict" what programs need RAM. This can cause ISAS to slow down dramatically, making it almost useless in some cases. This is all preventable by setting swappiness to zero.

ISAS can be run in interactive mode, or in production mode. When ISAS is run for the first time with no input arguments, it enters interactive more. It will load the reference into memory, and await further commands. You can interactively issue different commands (one by one) to get information about your hardware, reference, and settings, or to change settings, or to perform alignment on files. When you quit your interactive session, the settings are saved, to be reloaded the next time ISAS is run (either in interactive or production modes).

In production mode, ISAS is run (from the operating system command line, or from your calling script) with a single argument which represents a command. ISAS will start, load the settings last saved, execute the single command from the command line, then save the settings (your command could have been to change some setting), and exit.

We suggest that you use interactive mode to get familiar with the different capabilities and characteristics of the system. At a later stage, you can use production mode in batch files or under control of an automated program.

Once in an interactive ISAS session, a command is issued to ISAS by typing one of the supported commands, often followed by "=" and then the list of parameters (if any), separated by commas. The ENTER key is pushed to begin processing the command. ISAS Commands are all case insensitive.

You can easily get a list of the known commands by typing "?", followed by the ENTER key. From this point on, we won't mention "followed by the ENTER key" anymore – it should be implicitly understood that a command must be terminated by ENTER.

The transcript of the above session is show below:

```
 Enter next command, or type "?" (and ENTER) for list of commands.

 ?

 Enter one of the following:

1.  "STATUS" Display current system mode/parameters.
2.  "SEQUENCE=x" where x is a string starting with 1 base (A,C,T, or G)
    followed by 24 colors (0,1,2, or 3) for a single search.
3.  "FILE=X" Process csfasta sequence input file X.
4.  "FILEQ=X,Y" Process csfasta file X with corresponding qual file Y.
5.  "FILES=x,y,min,max" Process paired sequence input files x and y
    with min and max base pair distances allowed between "AAA" pairs.
6.  "MODE=x" Set search mode. Type "MODE=?" for explanation of modes.
7.  "GLOBAL=x,y" Enable Global Extension with sequence length x bases, and
    total mismatches y. Use x=25 (with dummy y) to disable Global Extension.
8.  "VERBOSE=x" Specify output file format. x=0 for Non-Verbose output,
    x=1 for Normal, x=2 for Unique-Only, or x=3 for SAM.
```

```
9.  "RANDOM=x" Create x random 25mers from current genome. They are written
    to a file "RandomLengthX-Y.txt" and simulate read errors (Y).
10. "RANDOMS=x" Create a set of mate-pair files with x random pairs of
    25mers from current genome.
11. "LIMIT=x" For each sequence, stop searching after x matches are found.
12. "REF=x,y" View reference chromosome x position y.
13. "CHR=A,B" Set first chromosome to A and last chromosome to B. If no
    chromosomes use "CHR=1,1". Remember to convert X,Y,M, etc. to numbers.
14. "FASTA" Convert fasta files to reference files. Needs to be done once.
15. "MAKEBIN" Create database files from reference files.
16. "HARDWARE" Display information about the system hardware (CPU and RAM).
17. "FILTER=x" x=0 for No Filter (100% sensitivity). 1 (min) through 15
    (max) to filter sequences and tradeoff sensitivity for speed.
18. "DATABASE=x" Change reference database. Load it from the directory "x".
19. "READGROUP=w,x,y,z" For SAM output file. Adds an RG tag to lines and
    a header: @RG ID:w PU:x LB:y SM:z. READGROUP=0,0,0,0 removes ReadGroup.
20. "QUIT" or "EXIT" to exit.

 Enter next command, or type "?" (and ENTER) for list of commands.
```

The following is a description of each of the supported commands:

1. STATUS

   Use the STATUS command to find out the current settings. First, ISAS will report the first and last chromosomes (both 1 if no chromosomes), and total number of bases in reference. Next, ISAS will report the current Mode (see 5 below) and if there is a Global Extension (see 6 below). Next, Verbose or Regular output (see 7 below), and the value of LIMIT (see 9 below). Finally, the number of input sequences aligned at a time will be reported (see section V 4 on page 22).

2. SEQUENCE=x

   Although extremely inefficient, you can use ISAS to perform alignment on a single sequence, as opposed to a file with millions of sequences. With this command, ISAS will display a “mismatch analysis” between the input sequence and the reference, for each match found (up to LIMIT, which is the maximum number of matches searched for). An “x” will be shown for each mismatch, and if in “valid adjacent” mode (see 5 below) a “v” will denote the second base of a pair of valid adjacent bases. This command is useful for understanding why a particular sequence is considered a match in a particular mode. It is extremely inefficient in terms of execution time, since the same time is spent on overhead as if many millions of sequences were being aligned. As discussed in section I 3, this can be 5 minutes on a computer with a fast hard disk system for a human reference genome., or 20 minutes for the same run on a computer with a slow hard disk. The “x” in the “Sequence=x” syntax represents the sequence. The format of the sequence is one base (“A”, “C”, “T”, or “G”) followed by ReadLength color codes. Each color can be “0”, “1”, “2”, or “3”. If the length of the entered sequence x is too short, as determined by the current settings, ISAS will reply with an error message indicating this problem. If x is too long, it will be truncated to the proper length, and a reply will contain a warning as to the truncation. Below is an example session transcript, the mode was 2VA (see section 5 below) which includes “valid adjacent”

color mismatch pairs, the Read Length was 35, and the total allowed mismatches was 3 (See section 6 below).

```
Enter next command, or type "?" (and ENTER) for list of commands.

sequence=T0102002133222032120230310231102l321

L0b 54.8 L1b 50.6 L2b 53.9 L3b 51.7 L4b 53.9 L5b 52.1 L6b 52.8
L7b 58.1 L8b 59.4 L9b 55.2 L10b 51.6 L11b 52.7 L12b 57.1

One match found. 703.8 Sec. (703.8 Sec plus 30861.0 micro Sec.)

6_1000000.3

       T102002133222032120230310231102l321
       T102002313222032120010310031102l321
              XV          XV    X           3 substitutions
```

The sequence entered, despite looking longer, is of length 35. The format used by the sequencer is perhaps slightly wasteful. The first base is described as “T0”, which is equivalent to “T”. ISAS uses a more compact format with the first base followed by Read Length – 1 color codes. The sequence was found, exactly in one location in the human genome: chromosome 6, genomic position 1,000,000, forward strand, and with a total of 3 mismatches. This is denoted in compact form as “6_1000000.3”. The input sequence is shown directly above the reference at the matched location, and any mismatches are indicated by “X” below them. The “V”s indicate valid adjacent pairs which are counted as single mismatches. Even though there are 5 base positions which don’t match between the input sequence and the reference, because we are in VA (Valid Adjacent) mode, we can determine that they are actually two SNPs (Single Nucleotide Polymorphisms) plus one machine sequencing error, for a total of 3 substitutions, which is within the maximum allowed of 3. If we run the same command in Mode=2, the sequence will not be found, as this reference location will count as 5 substitutions. If you are not yet familiar with color space and the concept of Valid Adjacent pairs, we recommend reading section VI and revisiting this example with a full understanding.

The row starting with “L0b…” provides some information that is helpful for performance analysis, specifically for your hardware. But what is interesting is that it took 704 seconds to run just this one sequence. The single sequence alignment is extremely inefficient, as we could have also aligned 1,000,000 sequences in about the same time, using multiple sequence alignment (the next command, 3 below).

3. FILE=x

Use this command to perform alignment on a file of sequences. “x” denotes the file name, including the full path. The file should be in “csfasta” format. A “header” made of lines beginning with the character “#” is allowed at the beginning of the file. The header, which will be ignored, should be followed by pairs of lines, one pair for each sequence.

The first line of each pair is the “Tag” line. The line must begin with the character “>” and it will be followed by whatever information we want to attribute to the sequence. Usually this is the ID that the sequencer assigned it. This line will be echoed in the output file, followed by the mapping results. The mapping results will show all the matches found (if any) separated by commas. Each match will contain the chromosome number (if any), and underscore (“_”), the genomic position within the chromosome, a period (“.”) and the number of mismatches. Reverse strand locations will be denoted by a negative genomic position.

The second line of each pair is the actual sequence, composed of one base (“A”, “C”, “T”, or “G”) followed by ReadLength color codes. Each color can be “0”, “1”, “2”, or “3”. If the length of the sequence is too long, as determined by the current settings, ISAS will ignore the extra bases. If it is too short it can create a problem for aligning, so you should take care not to try to align a file of 25mers as if they were 35mers. This line will be echoed as-is to the output file. The following is an example of an input file.

```
# This data was produced on sequencer no. 3 with human sample no. 4
# Time of sequencing: 12/12/08 4:41PM
# Files backed up at /home/databackup/sequencer3/21354
>469_26_42_F3
T12113310031232112221003120021221223320222122212122
>469_26_379_F3
T31202223003310000130302323312223212011000010033200
>469_26_540_F3
T11012313031030123033113130100223110001231232303210
```

ISAS will create a name for the output file that will capture the important aspects of the settings used to perform the alignment. It will start with the name of the input file, and then concatenate the current mode, read length, and, if a Global Extension (see 5 below) is used, it will also concatenate the setting for the Global Extension. The output verbosity type and input filter level will also be specified in the output file name. Finally, “.txt” will be added at the end.

The location of the output file will be the same as that of the input file. Make sure you have enough disk space when running this command. Output files are somewhat larger than their corresponding input files, as they contain the sequence and tag from the input file, plus the alignment results. For high LIMIT values, the output file size can become very much larger than the input file size, as potentially many non-unique match positions can be listed for sequences in repeat regions of the genome. Below is an example session transcript, the mode was 2 (see section 4 below) with no Global Extension (see 5 below), so the Read Length was 25.

```
Enter next command, or type "?" (and ENTER) for list of commands.

file=BARB/F3.csfasta

Opening output file BARB/F3.csfasta.Mode2.Length25.txt for writing...3.7 Sec.
I 1.3 L0a 50.9 R2 95.5 L1a 51.2 R2 30.0 L2a 54.2 R2 26.5 L3a 51.3 R2 72.2 L4a 50.7
```

```
R2 80.0 L5a 51.2 R2 72.1 L6a 52.0 R2 72.2 W39559955 77.4

Total 39559955 sequences done in 888.7 sec. (L=361.42 sec. R=448.43 sec. W=77.45 sec.)

Hits    Histogram   Percent
====    =========  =========
0       432402        1.1
1       26644904     67.4
2+      12482649     31.5

Enter next command, or type "?" (and ENTER) for list of commands.
```

As an added convenience, a histogram of the mapping results is displayed at the end. It shows how many times sequences were found, from zero (sequence not found in reference) to LIMIT+ (sequence was found LIMIT or more times). This is very useful for immediately identifying a problem, such incorrect settings or a bad experiment. For example, usually we expect most sequences to be found, then seeing almost all the sequences as not found immediately raises our suspicion, and we can look around quickly to find that we typed in the wrong Read Length, or maybe the wrong species was used as reference. Without the histogram, we would have to analyze Gigabytes worth of data to detect these simple human errors. Another example, where the histogram shows far too many sequences as having large number of repeated matches, draws our attention and we find that the DNA sample was not prepared properly, resulting in data that is mostly "." or "0".

This command will be used to illustrate "production mode" by including it in the command line when running ISAS. The settings established in the last "interactive mode" session will be used. For example, at the operating system prompt, typing:

./ISAScolors FILE=/home/solidFiles/HumanSample1.csfasta

Will start ISAS, load the reference files, load the settings from the last ISAS session, and perform alignment on the file "/home/solidFiles/HumanSample1.csfasta". After ISAS finishes writing the results file, it will exit. This mode allows running ISAS in an automated fashion, e.g. from scripts.

4. FILEQ=x,y

   Same as FILE=x (3 above) but with an additional quality score file y. This command is only useful when the output file is SAM format.

5. FILES=x1,x2,min,max

   Use this command to perform alignment on pairs of mated sequences ("mate pair"). "x1" denotes the name of the F3 file (including full path) and "x2" the name of the R3 file. The order is important, i.e. the F3 file name should come before the R3 file name). The format of these files is the same as defined for the "FILE" command (IV 3 above). "min" and "max" specify the minimum and maximum base pair distance expected between mated pairs of sequences.

ISAS will use the tags in both files to find corresponding pairs of sequences, where each pair shares the same tag, except for the endings “F3” vs. “R3”. Now the tags will have to observe the format “>x_y_z_F3” for the F3 (first) file, and “>x_y_z_F3” for the R3 (second) file. “x”, “y”, and “z” in the above tag format descriptions must be numerical. The input files must be sorted in ascending tag order, where “x” is most significant and “z” is least significant. Any sequence in the F3 file for which there is no corresponding mate in the R3 file will be ignored, and the total number of these sequences will be reported at the end of the run. Similarly for R3 sequences with no F3 mates.

Each sequence in a pair will be separately aligned, using the current mode. If matches are found for both sequences, pairs will be constructed and reported in the output file, based on the “Verbose” mode (see IV 7 below).

In modes 2 (with or without global extension), 3, 4, or 5, a second process called “Pairing” will be invoked. For each sequence pair, where one sequence was found (i.e. no. of matches greater than zero) but its corresponding mate sequence was not found (i.e. no. of matches equal zero), a second pass will search for the “missing” sequence with more lenient matching criteria. These new matches will be accepted only if their location relative to the known mate sequence falls within the specified mate pair criteria, i.e. same chromosome, same strand, correct order (F3 ahead of R3), and between “min” and “max” base pairs away from the known mate sequence. This pair location is called “AAA” and the requirement of no more than the maximum allowed mismatches will be for the average of the two mate sequences instead of individually.

For example, in mode 5 a maximum of 5 mismatches are ordinarily allowed for each 50mer. Therefore, the sum of the total mismatches across all 100 bases of two “AAA” mate pair sequences may not exceed 10. Under this example, if an R3 sequence was found at a particular location with a total of 3 mismatches (2 less than the maximum allowed), and its F3 mate sequence was not found with the regular search criterion (no more than 5 mismatches), then the Pairing process would allow for F3 sequences near the known R3 location with up to 7 mismatches (2 more than the regular maximum of 5).

In order to make use of the Pairing function, The maximum number of hits (see the LIMIT command in IV 9 below) should be set to values higher than 2. A value of 10 is commonly used.

The output of mate pair alignment will be recorded in an output file. The name of the file will reflect the various settings used at the time of the run, similar to the case of the “FILE” command (IV 3 above), but with the addition of the word “Paired” towards the end of the file name. Similarly a “Stats” file will be written summarizing the results. The format of the output file will depend on the “Verbose” setting. However, regardless of the Verbose setting, all output files will reflect only sequences that could be paired based on their tags from the F3 and R3 files. Any sequence from one input file whose tag ID could not be found in the other input file will simply be skipped.

In NonVerbose setting (Verbose=0) the output will be very simple. Each line will show the number of pair locations found for each sequence pair. This can be useful for easily producing histograms or other statistics without using a lot of disk space. Pairs with no matches will have a “0”, pairs with a unique match will have a “1”, etc.

In Regular setting (Verbose=1), each pair will contain the following information (separated by tabs): Tag ID, F3 input sequence, R3 input sequence, no. of mismatches for F3 match, no. of mismatches for R3 match, sum of both mismatches, chromosome for F3, chromosome for R3, position for F3, position for R3, and mate-pair category. If there are more locations where this mated pair sequences was found, each one will be represented by another line, but the tag and sequences will not be repeated in those following lines. Pairs with no matches will not be shown in the output file.

In Unique-Only setting (Verbose=2), only pairs with unique matches (i.e. exactly one location) will be shown. Pairs with no matches, or with multiple matches will be omitted.

The mate-pair category codes shown as the last column in Regular or Unique-Only output files are defined as follows:

AAA Same chromosome and strand. Correct order and relative distance.
AAB Same chromosome and strand, Correct order but distance less than “min”.
AAC Same chromosome and strand, Correct order but distance greater than “max”.

BAA Same chromosome but different strands. Correct relative distance.
BAB Same chromosome but different strands. Distance less than “min”.
BAC Same chromosome but different strands. Distance greater than “max”.
ABA Same chromosome and strand. Reverse order. Correct relative distance.
ABB Same chromosome and strand. Reverse order. Distance less than “min”.
ABC Same chromosome and strand. Reverse order. Distance greater than “max”.
C** Different chromosomes.

6. MODE=x

This command is used to set the current search mode. Potentially following some period of research, most users usually end up with a single preferred mode. ISAS will save the current mode upon exiting, and re-load it the next time it is run, saving you the “effort” of typing this command every time. Also, for your convenience, a list of all the available modes, with explanations, can be viewed by using the command “MODE=?".

The following modes are currently supported:

MODE=0
First 25 bases must have 0 mismatches (perfect match).

MODE=1
First 25 bases can have up to 1 mismatch (i.e. 0 or 1).

MODE=2
First 25 bases can have up to 2 mismatches (0, 1, or 2).

MODE=2VA
First 25 bases can have up to 2 mismatches, where two valid adjacent mismatches (VA) count as 1 mismatch.

MODE=02
First find all perfect matches for first 25, then add all locations with up to 2 mismatches for first 25.

MODE=012
First find all perfect matches for first 25, then add all locations with 1 mismatch for first 25, then add all locations with 2 mismatches for first 25.

MODE=02VA
First find all perfect matches for first 25, then add all locations with up to 2 mismatches for first 25 with VA pairs counting as singles.

MODE=012VA
First find all perfect matches for first 25, then add all 1mismatches for first 25, then add all 2VA mismatches for first 25.

MODE=3
First 50 bases can have up to 3 mismatches (0, 1, 2 or 3).

MODE=3VA
First 50 bases can have up to 3 mismatches, where two valid adjacent mismatches (VA) count as 1 mismatch.

MODE=4
First 50 bases can have up to 4 mismatches (0, 1, 2, 3 or 4).

MODE=4VA
First 50 bases can have up to 4 mismatches, where two valid adjacent mismatches (VA) count as 1 mismatch.

MODE=5 First 50 bases can have up to 5 mismatches (0, 1, 2, 3, 4 or 5).

In any of the "first 25" modes (all modes except 3, 3VA, 4, 4VA, and 5), the Read Length can be extended beyond 25 by using the command "GLOBAL=" (see 6 below).

In any mode, searching for any sequence stops after a total of LIMIT matches are found for that sequence. LIMIT can be changed using the “LIMIT=” commands (see 9 below).

7. GLOBAL=x,y

As explained in 5 above, the first eight modes perform alignment using exactly 25 bases (1 bases plus 24 colors), and the last five modes do so on 50mers. But by enabling Global Extensions, we can align sequences of lengths 33 through 62. Furthermore, Global Extensions are designed to alleviate the problem associated with read lengths greater than 25 - the reliability of the reads deteriorates the further

the base is along the read. Bases beyond the 25th position are significantly less reliable than the first 25 bases. When Global extensions are enabled, two standards of reliability can be employed, one for the first 25 bases, and a second (looser, if desired) for the remaining "extension" beyond 25.

While the current mode still applies to the first 25 bases in the case of any of the first 8 modes (the last 4 modes are not applicable with Global Extensions), another constraint on the total number of mismatches for the entire sequence is added when Global Extensions are enabled.

In the command syntax, "x" is the total Read Length, and "y" is the total number of mismatches allowed over the entire sequence. For example, assuming MODE=2, then the command "Global=35,4" will set the Read Length to 35, and define a match as follows:

First 25 bases of input sequence have no more than 2 mismatches with the reference.
AND
All 35 bases of input sequence have no more than 4 mismatches with the reference.

In this example, if the first 25 bases have 2 mismatches between the input and reference sequences, and the next 10 bases have another 2 mismatches between

input and reference, this location in the reference would be considered a match, as the total of 4 is within the Global allowance of 4. Note that allowing 2 mismatches over the last 10 bases is allowing 2½ times more "mismatches per base" than the criterion of 2 mismatches over the first 25 bases.

Using a value of 25 for "x" (with a dummy value for "y") will disable Global Extensions and return to 25mers. For example:

GLOBAL=25,5
Will disable Global Extensions. The Read Length will return to 25. The "5" will be ignored, and the number of mismatches allowed for the first 25 bases will be determined by the current search Mode.

Note: If the current search mode includes "valid adjacent" (VA), then the Global extension will also count valid adjacent pairs of mismatches as a single mismatch anywhere in the sequence (first 25 bases, as well as the extension beyond the first 25).

As with the mode, this setting will be saved when ISAS exits, and re-loaded when run the next time. So once you have "comfortably settled" on the search Mode and/or Global Extension settings that are best for your data, you won't have to type these commands again.

8. VERBOSE=x

Use a value of 1 for "x" to enable "Normal Output". In this state the output file will be as described in 3 above. A value of 0 for "x" will disable "Full Output". In this "Non Verbose Output" state, the output file will only report the number of matches for each sequence. The tag and the sequence itself will not be echoed to the output file. This can be useful for research purposes when comparing the effects of different modes and settings. By comparing two "Non Verbose" output files

produced from the same input file, but with different settings, specific sequences whose mapping characteristics have changed can easily be pin-pointed. Use a value of 2 for "UniqueOnly". In this setting, only sequences which are uniquely mapped (i.e. exactly one match) will be output. VERBOSE=3 will write the output file in "SAM" format. More information about this format, along with free software to perform various functions including downstream analysis on this type of file, can be obtained at http://samtools.sourceforge.net. To use "pileup" or SNP calling functions, make sure that the first line of each reference file has the numerical name, for example, if "chr23.fa" starts with the line ">chrX" it should be changed to ">chr23".

9. RANDOM=x

   This command is used for creating files of sequences for testing and research. "x" specifies the number of sequences. Each sequence will first be copied from a random location in the reference genome. Next, random mismatches will be added to the sequence, as defined by the current mode or Global Extension setting. The file name will begin with "Random", show the Read Length, total mismatches allowed (if Global Extension is enabled), search Mode, and end with ".txt". The format will comply with the csfasta format described in 3 above. instead of an ID, the tag line will contain the "correct" alignment information for the sequence, i.e. the chromosome (if any) and genomic position in the reference where it was originally copied from. A negative position denotes a reverse strand.

10. RANDOMS=x

    This command is simiar to RANDOM (9 above) except that it creates two mate pair files.

11. LIMIT=x

    Some sequences could be repeated hundreds of thousands of times within the reference genome. Normally, it would be a waste of CPU time and disk space to search for and record all these locations. Furthermore, if we consider the alignment function is to uniquely identify the location of each sequence, then it would be most efficient to stop searching after two matches are found. On the other hand, mated pairs of sequences can sometimes be "rescued" from multiple matches of each sequence to a unique match of the pair. This is why we allow the user to set LIMIT, which is the maximum number of matches searched for. "LIMIT=x" will set the LIMIT to "x". For each sequence, searching will stop if x matches are found. It is recommended to keep LIMIT as small as is practicable for your application. For single files, keep it at 2, and for pairs ("mate pair") of files use 10, unless your application requires finding more matches. Setting LIMIT to values higher than 2 will incur a performance degradation. If your system does not have enough RAM, then large values can be very costly in this sense (this is discussed in section V 5 below).

12. REF=x,y

    This function is very useful for research, and to investigate alignment behavior. ISAS will display the reference at chromosome "x" position "y". The lower row will

show the original “base space” reference, and the upper row will show the interleaved “color space” codes between each pair of bases of the reference. If the species has no chromosomes, use the value 1 for “x”. Reverse strands are denoted by negative values for “y”. A session transcript with this command can be seen in Section VI.

13. CHR=x,y

    Use this command to set the first chromosome to “x” and the last chromosome to “y”. If no chromosomes, use the value 1 for both “x” and “y”. For human, use “CHR=1,25” to include mitochondrial DNA, or “CHR=1,24” to exclude it. Normally this command only needs to be used once when setting up a new reference genome. ISAS will save this in the settings file when exiting, and re-load it when started the next time.

14. FASTA

    Convert fasta files to reference files. This command only needs to be used once per new reference genome. You don’t need it for the human reference genome as the reference files are included in the ISAS installation DVD for your convenience (see section III 2), unless replacing with a newer reference.

15. MAKEBIN

    Create database files from reference files. This command only needs to be used once per new reference genome, or twice for two databases: one for regular searches, and a second for “valid adjacent” searches (See sections III 1 and III 2).

16. HARDWARE

    Display information about system hardware. ISAS will interrogate the hardware it is running on, and report the number of logical CPUs, amount of RAM, and the capabilities of the CPUs. This is a generally useful function, as not only is it easier than looking up your system documentation, it is also a more reliable source of current system information, as the system could have been changed from the last time it was documented (e.g. RAM added or removed, or CPUs upgraded).

```
Enter next command, or type "?" (and ENTER) for list of commands.

hardware

CPU: Intel(R) Xeon(R) CPU E5335  @ 2.00GHz (Stepping=7 Model=15 Family=6)
No HyperThreading. No PC. Has SSE3-SSE3 Extensions.
Has SSSE3-SSSE3 Extensions. No SSE4.1. No SSE4.2.
8 logical processors, 25,281,826,816 bytes RAM.

Enter next command, or type "?" (and ENTER) for list of commands.
```

17. FILTER=x

    With FILTER=0, ISAS performs “lossless” alignment, also called “exhaustive” alignment. This means that no shortcuts are taken. If ISAS determines that there

are X matches for a sequence, then there are exactly X locations in the reference which match the sequence within the maximum allowed mismatches. The only possible exception is when X reaches LIMIT. Filtering can optionally be used to further increase speed, by applying shortcuts to “problematic” sequences that are likely to be in repeat regions. Ten levels of filtering are available, from FILTER=1 (very slight filtering) to FILTER=10 (more aggressive filtering). Our experience (for human reference) has shown that using FILTER=3, alignment speed can be doubled by losing less than 0.1% of the sequences.

18. DATABASE=x

    In its default, installed state, ISAS expects to be using the human genome reference known as “hg19”. This means that in the same directory where the ISAS executable (program) resides, there will be a sub-directory called “hg19”. This directory will hold the settings file, which stores your preferred settings (Mode, Verbosity, LIMIT, Global, etc), as well as the reference database files, and a subdirectory called “reference”, which holds the reference files. The “DATABASE” command is used to switch to a different reference. The current reference is stored in a file called “ReferenceDirectory.txt” so when ISAS starts up, it will know which database directory to load. For example, to switch to the mouse reference mm9, type:

    DATABASE=mm9

    Note that in Linux, directory names are case sensitive. So the name of the directory cannot be “MM9”, it must be “mm9”. The structure and content of the directory “mm9” must be similar to the “hg19” directory, just with mouse data instead of human.

19. READGROUP=w,x,y,z

    This command sets the “RG” (Read Group) values used in writing SAM files. The following header will be added to the SAM output file:
    @RG ID:w PU:x LB:y SM:z
    and each line will also contain an RG tag with the value w. To disable (remove) the RG tag, use the command READGROUP=0,0,0,0. This command is only useful when writing SAM format output files.

20. QUIT

    Terminate the program. Just before terminating, the values of the following settings will be saved in a file called “settings-colors.txt”:

    FirstChromosome
    LastChromosome
    Search Mode
    LIMIT
    Global Extension State (enabled or disabled)
    Read Length (if Global Extension is enabled)
    Total Allowed Mismatches (if Global Extension is enabled)
    Verbose Level
    Filter Level
    Read Group

This file will be in the current database directory. The current database directory name will be saved in the file "ReferenceDirectory.txt" in the executable directory.

These files will be read back the next time the program is run.

21. EXIT

Same as "Quit" (16 above).

V. Performance Issues

1. BIOS Setting - Prefetch

IMPORTANT PERFORMANCE NOTE FOR INTEL XEON 53xx and 54xx CPUs

Ignoring this will result in 40% slower alignment times.
On Xeon 53xx and 54xx CPU systems, turn off "Hardware Prefetch" and "Adjacent Cach Line Prefetch" in your BIOS settings.

Some computers are shipped with a default BIOS setting of "hardware prefetch" and "Adjacent Cache Line Prefetch" enabled. This causes the Xeon 53xx and 54xx CPUs to run ISAS (and many other programs which are memory intensive) 40% slower. If so, disable both of these in the BIOS settings utility program that you can invoke when you turn the power on. For example, in the PHOENIX BIOS utility program, these are in the "Advanced" tab, under "Advanced Processor Options". We encountered this issue with many customers, so this is a common problem in popular brands like HP, Sun, and Dell. You should ignore this issue for all AMD CPUs, or for more modern Intel CPUs.

2. Multi-Tasking

IMPORTANT PERFORMANCE NOTE

When running alignment – do not run any other program that might access the hard disk or RAM on the same computer.

Although modern computers are designed with the goal of multi-tasking, they usually are not capable of sharing their hard disk among multiple programs in a timely manner. CPUs can be "switched" back and forth between different programs with a switching overhead that can be reasonable if implemented correctly. However, the hard disk is a mechanical device, and when multiple programs are reading and/or writing on the same hard disk, the hard disk's head needs to physically jump back and forth between the two programs' desired locations. If a second program is trying to read or write a large file at the same time ISAS is reading a large file (which it often does), this can cause both programs to run **many times slower**.

3. Virtual Memory (Swapping)

If you don't have the minimum amount of RAM (see section I 2) required, but still try to run a program, then the computer will try to substitute hard disk for RAM. This process is called "swapping" or "virtual memory". Hard disk is orders of magnitude slower than RAM, and the overhead of swapping back and forth large parts of memory between RAM and hard disk for every small memory transaction can be overwhelming for the computer. The result is that the use of virtual memory causes programs to run **many times slower**, often causing the system to come to a virtual standstill. The reason virtual memory is used is that this situation is sometimes considered better than the alternative – which is a program crashing due to insufficient memory. However, in non-critical applications, it may be better to crash a program than to drag down the entire computer. In any case, it is not advisable to use ISAS with a reference genome which requires more RAM than you have. If you do, you must expect dramatic performance degradation. In addition, the Linux swapping algorithm will consider any RAM that has not been used for several minutes to be "useless", and can swap this memory to hard disk even when there is enough RAM and there is no need to do this. Since alignment takes more than a few seconds, this can happen, and then the Linux swapping mechanism will actually destroy the fast performance of ISAS. This is why it is important to set the Linux "swappiness" parameter to 0 (zero). We recommend permanently setting it to 0, as this is probably the most efficient way to operate servers. See section III 1 on page 5 for instructions.

4. RAM and Multiple Sequence Alignment

Assuming your system has the minimal required RAM for your reference genome (explained in section I 2), you will avoid the "wrath" of virtual memory (see section V 3). The more extra RAM beyond the minimum requirement, the more sequences can be efficiently aligned simultaneously. We call the number of sequences that are aligned simultaneously the "buffer size". If your buffer size is 317 million sequences, then any file containing up to 317 million input sequences will incur the same overhead to align. Files containing more sequences, up to 634 million sequences, will incur double this overhead. While the exact numbers vary based on your computer, the reference genome, and ISAS run time settings, we will use an example to illustrate the effect of RAM size on buffer size, and thus on multiple sequence alignment speed.

Assume a human reference, ReadLength=50 bases, Limit=2, 24GB of RAM, and a multiple sequence alignment overhead of 12 minutes. Based on this scenario, your current buffer size is 317 million sequences. If you run a file with 318 million sequences (1 million sequences more than your buffer size) it will take 12 minutes more than a file with 317 million sequences. The alignment of the extra 1 million sequences is only a few seconds, but an additional 12 minutes of overhead are incurred for exceeding the buffer size. If you align a file with 634 sequences, it will take twice as long as the 317 million sequences file. Next, assume that you add 4GB more RAM to your computer (from 24GB to 30GB). Now, your buffer size becomes 456 millions sequences. The 318 million sequences file will now take only a few seconds longer to align than the 317 million sequences file, and it will take twice as long to align 912 million sequences as compared with 456 million sequences.

As a rule of thumb, if you have 24GB of RAM, and use a Limit of 2, or 32GB RAM with a Limit of 10 or less, you can expect good performance on human genome. On small organisms, you probably don't need to care about RAM.

5. LIMIT (Maximum No. of Matches)

For each sequence, ISAS will keep searching the reference database until either there are no more matches (within the specifications of the current mode), or LIMIT matches have been found, whichever comes first. Normally, setting LIMIT=2 is all that is required to determine if a sequence can be uniquely mapped to the reference. Higher values for LIMIT will cause slightly slower alignment times, and greatly reduce the multiple sequence alignment buffer size (see section V 3 above) which can degrade greatly increase total alignment time. Why would a LIMIT value higher than 2 be used ? Multiple candidate locations for sequences that cannot be uniquely mapped might be of interest for some applications or research. Another use is "mate pair" or "paired" sequencing. In this situation, the sequencer produces pairs of sequences, where the relative genomic locations of the pairs is known (within some tolerances). Examine the example below.

Mate Pair "Rescue" Example:
It is known that each pair of sequences is spaced 1K bases +/- 200 bases apart from each other. We can perform alignment with LIMIT=10. Even if both sequences of a pair are not uniquely mapped, but both of them are found anywhere from 1 to 9 times, then we can usually perform a process of elimination based on the constraint on the genomic distance between the two sequences in the pair to uniquely map them.

VI. Tutorial: Color Space and Valid Adjacent Pairs

Performing alignment in VA ("Valid Adjacent") mode takes twice as long as regular alignment, and twice as much disk space is required. Is it worth it ? This section explains the merits of VA modes, unique to color space sequencing.

In color space based sequencing, the sequencer reports "color codes" instead of bases. Each such code represents the "difference" between adjacent bases. It is useful to view both bases and colors interleaved together.

```
Enter next command, or type "?" (and ENTER) for list of commands.

ref=6,1000000

chr 6 pos 1000000:  1 0 2 0 0 2 3 1 3 2 2 2 0 3 2 1 2 0 0 1 0 3 1 0 0 3 1 1 0 2 1 3 2 1
chr 6 pos 1000000: T G G A A A G C A T C T C C G A C T T T G G C A A A T G T T C A T C A
```

Above is a session transcript from ISAS. In this session, ISAS is used as a "reference lookup", we are inquiring the reference sequence at chromosome 6, starting at genomic position 1,000,000 (base pairs). The current Read Length is 35, so a sequence of length 35bp starting at position 1,000,000 is shown. The lower row shows the 35 reference bases ("A","T","C","G"). The upper row shows 34 color codes "sandwiched" between the bases. Each code is the "difference" between the base on its left and the base on its right. For example, the first (left-most) code "1" is the difference between the "T" on its left and the

"G" on its right. The second code "0" is the difference between "G" and "G". Differences between bases are arbitrarily defined in the difference table below.

Base	Base	Difference
T	G	1
A	C	1
A	G	2
T	C	2
A	T	3
C	G	3

The order of the two bases is not important in calculating the difference (color code). Whether "T" is on the left and "G" is on the right, or whether "G" is on the left and "T" is on the right, the difference is "1" in both cases. Furthermore, the difference is "0" if the same base is on both sides. A very important property of the distance defined above is that if *exactly one* of the bases of the pair is changed, then the color code also always changes. This property will turn out to be is very useful.

All sequencers are prone to "machine" errors (sometimes called "sequencing" errors) due to various reasons (chemical, mechanical, electrical). The use of these "pair differences" can give us a certain layer of protection against sporadic machine errors. Let's revisit the example above (chr. 6, pos. 1,000,000), but under three new situations. In the first scenario, the sample being sequenced has a SNP (Single Nucleotide Polymorphism) occurring at position 1,000,002. That base (third from left) is now a "C" whereas the reference is a "G". In the second scenario, there is no SNP, but there was a single machine error which caused the sequencer to report a "0" instead of a "2" for the third color (from the left). In the third scenario, again there was no SNP, and again the sequencer had an error, but this time two color codes were corrupted: both the second and third colors were reported as "3".

	Scenario 1 One SNP	Scenario 2 One Error	Scenario 3 Two Errors
Sequenced Sample Bases	T G C A A	T G G A A	T G G A A
Colors Reported by Sequencer	1 3 1 0	1 0 0 0	1 3 3 0
Colors Computed from Reference	1 0 2 0	1 0 2 0	1 0 2 0
Reference Bases	T G G A A	T G G A A	T G G A A

We observe that the single SNP ("G" → "C") in Scenario 1 caused *two* color changes ("02" → "31"). The "0" (2nd color from left) was the distance between "G" and "G" in the reference. Now that the right base in this pair of bases "G" is a "C" in the input sequence, the "0" changed to a "3". Similarly, the "2" (3nd color from left) was the distance between "G" and "A" in the reference. Now that the left base of this pair "G" is a "C" in the input sequence, the "2" changed to a "1". Due to the property of distance to change when exactly one base changes, it becomes obvious that every single SNP (with no other SNP adjacent to it) will always cause an adjacent pair of color codes to change. The color code to the left of the SNP has (only) its right base changed, and the color code to the right of the SNP has (only) its left base changed. So, to be able to detect a single SNP, we need to allow a minimum of two substitutions from reference.

From what we learned from Scenario 1, we can deduce that the sequence shown in Scenario 2 is the result of a machine error, rather than a SNP. Of course, we are making

the assumption that there are no two adjacent SNPs. This assumption is not necessarily true, but two adjacent SNPs are so rare, that this assumption is useful for our purpose of detecting single SNPs without getting confused by machine errors.

But realistically, machine errors are much more common than SNPs, so we *cannot* assume that we will not encounter two consecutive machine errors. Fortunately, the "valid adjacent" technique can protect us from the vast majority of these cases, too. In Scenario 3. There is no real SNP, but rather two consecutive machine errors ("02" → "33") at the 2nd and 3rd color codes, which might impersonate a SNP at the 3rd base. We will examine all the possibilities (again, under the assumption of no two consecutive SNPs). Possibility 1 assumes a SNP whereby the 3rd base is an "A". Possibility 2 assumes the SNP is a "C". The 3rd possibility assumes a "T" SNP, and the 4th possibility is that there is no SNP in the sequenced sample, but two consecutive machine errors caused the "02" → "33" difference between reference and input sequence.

T G A A A	T G C A A	T G T A A	T G G A A
1 3 3 0	1 3 3 0	1 3 3 0	1 3 3 0
1 0 2 0	1 0 2 0	1 0 2 0	1 0 2 0
T G G A A	T G G A A	T G G A A	T G G A A
Possibility 1	Possibility 2	Possibility 3	Possibility 4

A closer look at Possibility 1 shows inconsistencies with our assumption of no two adjacent SNPs, or there have to be many more machine errors. The color to the left of the presumed "A" SNP (3rd base) is "3" (2nd color). Examining the difference table on page 24, it is clear that the base directly to the left of the "A" must be a "T" (2nd base), but if there are no two consecutive SNPs, it should be a "G", just like the 2nd base of the reference. Similarly, regarding the 4th base, i.e. immediately to the right of the presumed SNP: one base "A" and a color of "3" implies that the other base in the pair must be "T", but we know that the 4th base must be an "A", just like the reference, otherwise the "no two adjacent SNPs" assumption is violated. So, in order for the 3rd base to really be a SNP, it would require that there were more SNPs present (immediately to the left or to the right), or many more machine errors. Of these possibilities, it is orders of magnitude more likely that there are no SNPs present, and this is simply a case of two adjacent machine errors.

Possibility 2 will play out similarly. "C" + "3" = "G", but the 2nd and 4th bases should be "G" and "A" respectively. Possibility 3 is also inconsistent: "T" + "3" = "A" which work for the right side of the presumed SNP (4th position) but is inconsistent on the left side, where the 2nd base should be a "G". Finally, by elimination, we reach the conclusion that Possibility 4 is, by orders of magnitude, the most likely. The sequencer made two "mistakes", the 2nd and 3rd colors have been corrupted, as they should have been "0" and "2", respectively.

The reason we were able to "catch" the two machine errors, is that "33" and "02" are not Valid Adjacent (VA) pairs, as opposed to "31" and "02" which *are* VA pairs. VA simply means that if we work out all the possibilities, we will find one that makes sense. Not VA means that if we work out all the possibilities, none of them will make sense. After working out all the combinations for all pairs, we can create a table of all VA pairs:

00 11 22 33
30 12 21 03
20 02 31 13
32 10 23 01

Each row contains four pairs. Any of these four pairs is VA with any of the other three pairs in the same row. Looking back at the examples we had worked out, we see that "31" and "02" are on the same (3rd) row, thus they are VA, and this is the "true' SNP from Scenario 1, while "33" and "02" are on different rows (1st and 3rd), which means they are not VA, and not a true SNP, as we saw in Scenario 3.

What we've seen is that intelligent use of color codes can discriminate between a single machine error, a true single SNP, and even a pair of adjacent machine errors. While a good consensus algorithm for detecting SNPs (downstream from alignment) should take advantage of this, it would be handicapped if the alignment did not. We designed the VA search Modes in ISAS to take advantage of this. While in Mode=2 we are guaranteed to find sequences with up to 2 mismatches, just one SNP will exhaust the "mismatch budget", as it will cause two color code mismatches. If one machine error occurs in this read (and with readlengths of 25 to 50 this not unlikely), the total of 3 mismatches would cause the sequence to not be found when compared to the reference. If discovering SNPs is the goal, then Mode=2VA would solve this problem. In this search Mode, The true SNP would cause 2 valid adjacent mismatches, which would be counted as a single mismatch. Even after adding a machine error somewhere else in this sequence, the total would be 2 mismatches, guaranteeing that ISAS will find this sequence in the reference. In this case, it would be worth to have to take twice as long for alignment to complete, as the true benefit of the color code scheme would come to light.

Armed with a complete understanding of color space, and the concept of Valid Adjacent mismatched pairs, you may want to revisit the example in Section IV 2, and appreciate the added value of the VA capabilities of ISAS.

VII. Imagenix Contact Information

The ISAS team at Imagenix has attempted to design and create a useful, easy to use product. Our hope is that it will improve your productivity, and may be one of many stepping stones to help you obtain results which will benefit all of mankind. However, we realize that there is *always* room for improvement, and we always welcome feedback from our customers.

If you have questions, comments, or suggestions of a technical nature, that you would like to share with us, we have found that the most effective means for this is generally by email, as it allows both sides time to think before replying. It is helpful if you copy and paste the session text from your console or terminal into your email.

For technical support or feedback	technicalsupport@imagenix.com
For licensing information	sales@imagenix.com
General Information and updates	www.imagenix.com/genomics
Imagenix Technologies 171 Main Street #108 Los Altos, CA 94022	Tel. (650) 917-9998 Fax (650) 917-8765