# Detecting copy number variation with mated short reads

Paul Medvedev, Marc Fiume, Misko Dzamba, et al.

*Genome Res.* published online August 30, 2010
Access the most recent version at doi:10.1101/gr.106344.110

Resource

# Detecting copy number variation with mated short reads

Paul Medvedev,[1] Marc Fiume,[1] Misko Dzamba,[1] Tim Smith,[1] and Michael Brudno[1,2,3]

[1]*Department of Computer Science, University of Toronto, Toronto, Ontario M5R 3G4, Canada;* [2]*Banting and Best, Department of Medical Research and Centre for Analysis of Genome Evolution and Function, University of Toronto, Toronto, Ontario M5R 3G4, Canada*

The development of high-throughput sequencing (HTS) technologies has opened the door to novel methods for detecting copy number variants (CNVs) in the human genome. While in the past CNVs have been detected based on array CGH data, recent studies have shown that depth-of-coverage information from HTS technologies can also be used for the reliable identification of large copy-variable regions. Such methods, however, are hindered by sequencing biases that lead certain regions of the genome to be over- or undersampled, lowering their resolution and ability to accurately identify the exact breakpoints of the variants. In this work, we develop a method for CNV detection that supplements the depth-of-coverage with paired-end mapping information, where mate pairs mapping discordantly to the reference serve to indicate the presence of variation. Our algorithm, called CNVer, combines this information within a unified computational framework called the donor graph, allowing us to better mitigate the sequencing biases that cause uneven local coverage and accurately predict CNVs. We use CNVer to detect 4879 CNVs in the recently described genome of a Yoruban individual. Most of the calls (77%) coincide with previously known variants within the Database of Genomic Variants, while 81% of deletion copy number variants previously known for this individual coincide with one of our loss calls. Furthermore, we demonstrate that CNVer can reconstruct the absolute copy counts of segments of the donor genome and evaluate the feasibility of using CNVer with low coverage datasets.

[Supplemental material is available online at http://www.genome.org. CNVer is publicly available at http://compbio.cs.toronto.edu/cnver.]

The recent discovery of structural (Tuzun et al. 2005; Korbel et al. 2007) and copy number (Iafrate et al. 2004) genomic variation as a significant contributor to heterozygosity has revolutionized our understanding of the landscape of human genotypes. In the last few years several studies have characterized these variants in several human individuals (Bentley et al. 2008; Cooper et al. 2008; Kidd et al. 2008; Lee et al. 2008, 2009; McCarroll et al. 2008; Hormozdiari et al. 2009; McKernan et al. 2009; Yoon et al. 2009), demonstrating that of the total amount of variation between two human individuals, a larger fraction is in copy number and structural variants than in single nucleotide polymorphisms (SNPs) and other smaller scale variants. Structural variants (SVs), regions that contain insertions and deletions (indels) or inversions, and copy number variants (CNVs), regions appearing a different number of times in different individuals, are closely related phenomena, in that any change in copy number of a segment is a deletion or insertion that alters the structure of the genome. Nevertheless, the methods that have been used to characterize the two phenomena have been distinct.

Methods for CNV detection were until recently based on whole-genome array comparative genome hybridization (aCGH), which tests the relative frequencies of probe DNA segments between two genomes (for a comprehensive review, see Carter 2007). Alternate approaches have taken advantage of the extensive polymorphism data available from the HapMap project and used SNP arrays to measure the intensity of probe signals at known SNP loci (Cooper et al. 2008). However, while computational methods based on array data have been successfully used to identify CNVs, their power is limited. The size and breakpoint resolution of any prediction is correlated with the density of the probes on the array, which is limited by either the density of the array itself (for aCGH) or by the density of known SNP loci (for SNP arrays). Furthermore, the limited resolution of arrays for high-copy count segments and the paucity of unique probes make it difficult to identify CNVs in repetitive regions. These limitations have hindered the use of CNV predictions in association studies (Carter 2007; McCarroll and Altshuler 2007).

The advent of high-throughput sequencing (HTS) technologies has spurred new methods for CNV discovery. Current HTS datasets provide as much as 40× coverage for a human individual (called the donor), allowing the identification of regions with variable copy number by analyzing the depth-of-coverage (DOC): the density of reads mapping to the region. Several recent studies have shown that by comparing the DOC within a sliding window of the genome to what is expected in the reference genome, it is possible to detect changes in copy number (Campbell et al. 2008; Alkan et al. 2009; Chiang et al. 2009; Yoon et al. 2009; Simpson et al. 2010). This procedure, however, is complicated by the various biases of the sequencing process (Dohm et al. 2008; Harismendy et al. 2009), which make it difficult to separate true changes in copy number from segments that are over- or undersampled by the sequencing technology.

Alternatively, methods for SV detection have used clone-end sequencing data to detect variants (Raphael et al. 2003; Volik et al. 2003; Tuzun et al. 2005; Korbel et al. 2007; Bentley et al. 2008; Kidd et al. 2008; Lee et al. 2008, 2009; Hormozdiari et al. 2009). In this approach, two paired reads (called mate pairs) are generated at an approximately known distance in the donor genome. The reads are mapped to a reference genome, and mate pairs mapping at a

[3]**Corresponding author.**
**E-mail brudno@cs.toronto.edu; fax (416) 978-1455.**
Article published online before print. Article and publication date are at http://www.genome.org/cgi/doi/10.1101/gr.106344.110.

distance significantly different from the expected length (termed discordant) suggest structural variants. Paired-end mapping (PEM) techniques, which are based on the mining of such discordant mate pair information, have been successfully used for SV detection; however, they have difficulty detecting larger insertions and variation within areas of segmental duplications (see Medvedev et al. 2009 for a survey of methods for SV/CNV detection with HTS data).

While SV methods annotate regions which contain indels or inversions, CNV methods annotate the genome with loss/gain calls. Simultaneously, the events underlying these annotations are in many cases the same, and the two methods are often used to partially cross-validate each other (Tuzun et al. 2005; Campbell et al. 2008; Cooper et al. 2008; Kidd et al. 2008). For example, while inversions do not cause any changes in copy number, an area that is deleted (SV) will correspond to a loss (CNV). Similarly, a region containing a tandem duplication will be annotated as both having an insertion (SV) and as exhibiting a gain (CNV). In this way, any PEM method for SV detection can be viewed as a method for detecting a subset of CNVs. Beyond this trivial relationship, however, the methods for CNV and SV detection have remained distinct, and the full power of PEM has not yet been directly harnessed for the CNV prediction.

In this work, we show that PEM techniques can be used to improve both the sensitivity and specificity of DOC-based methods. Our method, called CNVer, supplements DOC information by mining discordant mate pairs in a way that specifically targets CNV detection. The DOC, PEM, and reference genome are all unified within a novel computational framework called the "donor graph"—an extension of the repeat graphs commonly used for genome assembly and alignment (Pevzner et al. 2004; Raphael et al. 2004; Paten et al. 2008). We use CNVer to detect CNVs within the NA18507 individual using a data set of 3.6 billion 36-bp-long reads. We make 2264 gain calls and 2615 loss calls (4879 total). Most of the calls (77%) coincide with previously known variants within the Database of Genomic Variants. At the same time, 81% of deletion CNVs previously known for this individual are identified by our method. Furthermore, we demonstrate that CNVer can reconstruct the absolute copy counts of segments of the donor genome and evaluate the feasibility of using CNVer with low-coverage datasets.

## Results

We first present a short intuitive view of our algorithm (fully described in the Methods section), followed by our results for a Yoruban individual sequenced with the Illumina technology (Bentley et al. 2008).

### Algorithms

Our algorithm considers two types of evidence that are indicative of CNVs within a probabilistic graph-theoretic framework. The first type of evidence is the distance information provided by paired-end mapping, which we use to infer adjacent regions of the donor's genome that have been separated in the reference. The group of mate pairs that span the breakpoint in the donor will map to the reference with a distance and/or orientation that differs from the expected. Taken together, these discordant mate pairs form a linking cluster (Fig. 1A), which we identify by grouping together discordant mappings that have a similar mapped distance, order, and orientation. These linking clusters are not associated with any particular type of variation, but simply identify adjacencies that may be present in the donor, but are absent in the reference. For example, a tandem duplication (Fig. 1B) will create a cluster that joins the end of the duplicated region to its beginning, while a deletion (Fig. 1C) creates a cluster that links together the bounding segments of the deleted region. Other clusters can link regions that are arbitrarily distant (e.g., nontandem duplications).

The second type of evidence is the DOC signature. Assuming the sequencing process is uniform, the number of reads sampled from a given region is proportional to the number of times the region appears in the donor's genome. A region that has been deleted (duplicated) relative to the reference will have less (more) reads mapping to it. This signal is independent of, and complementary to, the linking clusters, as is illustrated in Figure 1. These two types of evidence are then considered jointly by our algorithm within a framework that we term the donor graph, a data structure that is similar to the repeat graph (Pevzner et al. 2004).

When a read is sampled from a long segment appearing multiple times in the genome (e.g., a recent segmental duplication), it is impossible to discern from which copy it originated. We treat such segments in a unified manner by representing them together in a single edge of the donor graph, which is also annotated with the DOC on the corresponding segments. The edges are connected if their corresponding segments are adjacent either in the reference (which is known from the sequence) or in the donor (inferred from the linking clusters). The resulting donor graph can be understood as a compact representation of the donor genome.

In an earlier work, we showed that it is possible to identify the maximum-likelihood copy counts for each edge (segment) in a repeat graph based on the DOC and the graph structure by applying the network flow computational technique (Medvedev and Brudno 2009). Here, we extend this approach to work on donor graphs in order to reconstruct the maximum-likelihood copy count of segments in the donor genome. We compare these to the reference copy count to annotate regions of the reference as gain or loss CNVs. Our algorithm can also report the absolute copy count for any region. Note that while we can predict the regions that have been gained, we generally cannot identify the location where the duplicated sequence is inserted.

### Predictions and validation

We applied CNVer to the whole-genome shotgun-mated data set generated by Illumina for the Yoruban HapMap individual NA18507 with ~3.6 G reads, each ~36 bp long, and a mean insert size of 208 bp (Bentley et al. 2008). We ran CNVer on the autosomes, and made a total of 4879 CNV calls, of which 2615 were losses and 2264 were gains (Supplemental Table 1). Consistent with other studies (Korbel et al. 2007; Kidd et al. 2008), the number of our calls decreases exponentially as the length of the calls increases (Supplemental Fig. 1). Our calls cover a combined 1.34% of the autosomal genome. Segmental duplications have a density of called bases that is substantially higher than in other regions (21% vs. 0.4%). This corresponds with other studies, which also demonstrate a significant enrichment of CNVs within these areas (Locke et al. 2003; Iafrate et al. 2004; Sebat et al. 2004; Tuzun et al. 2005; Redon et al. 2006).

### Accuracy validation

In order to confirm the accuracy of our predictions, we compare them with CNVs previously identified for the NA18507 individual via wetlab and computational techniques, as well as to databases of previously known CNVs in a number of individuals. To test the

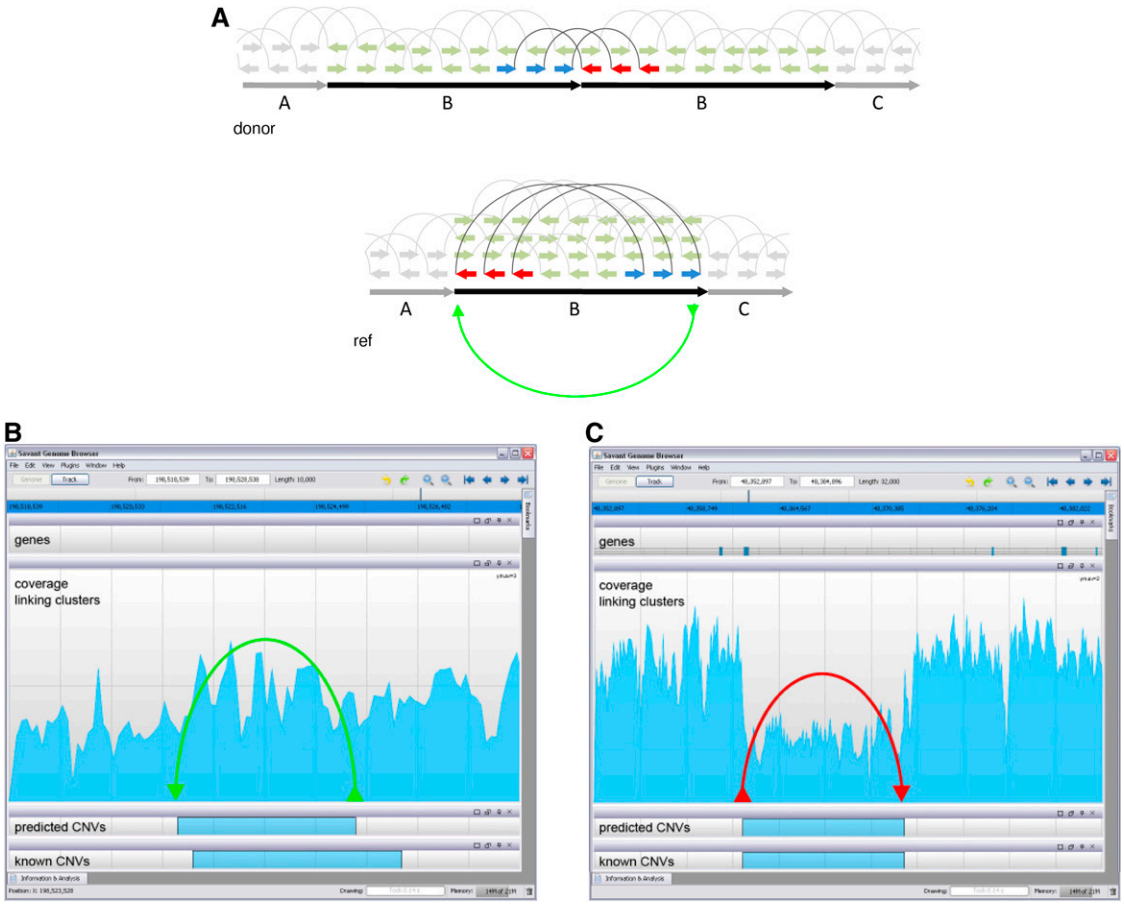


**Figure 1.** Depth-of-coverage and linking clusters. (*A*) A tandem duplication changes the DOC and creates discordant mate pairs. The reference genome consists of the segments ABC, while the donor has ABBC. The mate pairs are shown along the donor as they are sequenced and on the reference as they are mapped. The green (light) mate pairs are concordant, with the correct orientation and distance; however, the three bold mate pairs that span the adjacency between the two Bs are discordant. They form a linking cluster that indicates a putative adjacency in the donor from the end of B to the beginning of B (shown by the green edge). In addition, the duplication affects the DOC, as the density of reads mapping to the B segment is twice as high as in the rest of the genome. (*B*) A screen shot of the Savant (Fiume et al. 2010) genome browser (http://compbio.cs.toronto.edu/savant) on a 9-kb region of chromosome 1, showing the DOC and the linking clusters found by CNVer in our dataset, the gain call CNVer predicts, and an overlapping call from the GSV database of known variants. The underlying variation is likely a tandem duplication, like the one shown in *A*, because of the clear increase in the DOC and the presence of the green linking cluster. (*C*) A 19-kb region of chromosome 7 containing a deletion validated by sequencing (Kidd et al. 2008). There is a clear drop off in the DOC, as well as a linking cluster that connects the region *left* of the deletion to the one *right* of the deletion. Despite the noise in the DOC signal, the linking cluster allows CNVer to discern the breakpoints of the 10-kb deletion: they are predicted to within 28 bp on the *left* and 3 bp on the *right*.

sensitivity of our method, we measure its ability to identify deletion calls from Kidd et al. (2008), who used PEM of Sanger-style reads to identify indels in NA18507, which were then validated via aCGH. Since the mean and standard deviation size of their data set was large (37 ± 4 kb), their calls are generally overestimates of the actual variants. Therefore, we consider a call detected if we make a call that lies fully inside of it. In this way, we are able to detect 84% of Kidd et al.'s calls, with 96% of these containing a loss call (Fig. 2A). For the purposes of verifying the statistical significance of these comparisons, we shuffled our results by moving each of our predictions to a random location on the genome while maintaining its length and call type. We created 10 different shuffled versions of our results and repeated the comparison against Kidd et al. with each one of them. On average, only 12% of the Kidd et al. deletions were overlapped by one of the shuffled calls, with 59% of these overlapping a loss call.

Using the same data set as in this work, Bentley et al. (2008) used PEM to discover deletions within NA18507. Comparing against their deletion calls greater than 1 kb (1933 calls), we found that 47% of their calls overlap with ours, with 71% of these overlapping one of our loss calls (Fig. 2A). Against the shuffled data, only 3% overlap a call, with 53% of these overlapping a gain. When using a more stringent threshold of 50% reciprocal overlap, we found 32% of their calls overlap one of ours, with 86% of these overlapping a loss.

As most PEM-based methods, Kidd et al. (2008) and Bentley et al. (2008) predict the locations of insertions, rather than the original duplicated sequence, which makes it difficult to compare our gain calls against their insertions. We therefore compare our gain calls with the study of McCarroll et al. (2008), who used aCGH to identify genomic regions with a significant difference in intensity in a pool of 270 HapMap individuals (including NA18507). By further using the HapMap-wide intensity data for each variant locus, they estimated its actual (nonrelative) copy number in each individual. Because the human genome we use as our reference was not one of the genomes considered, calls that are

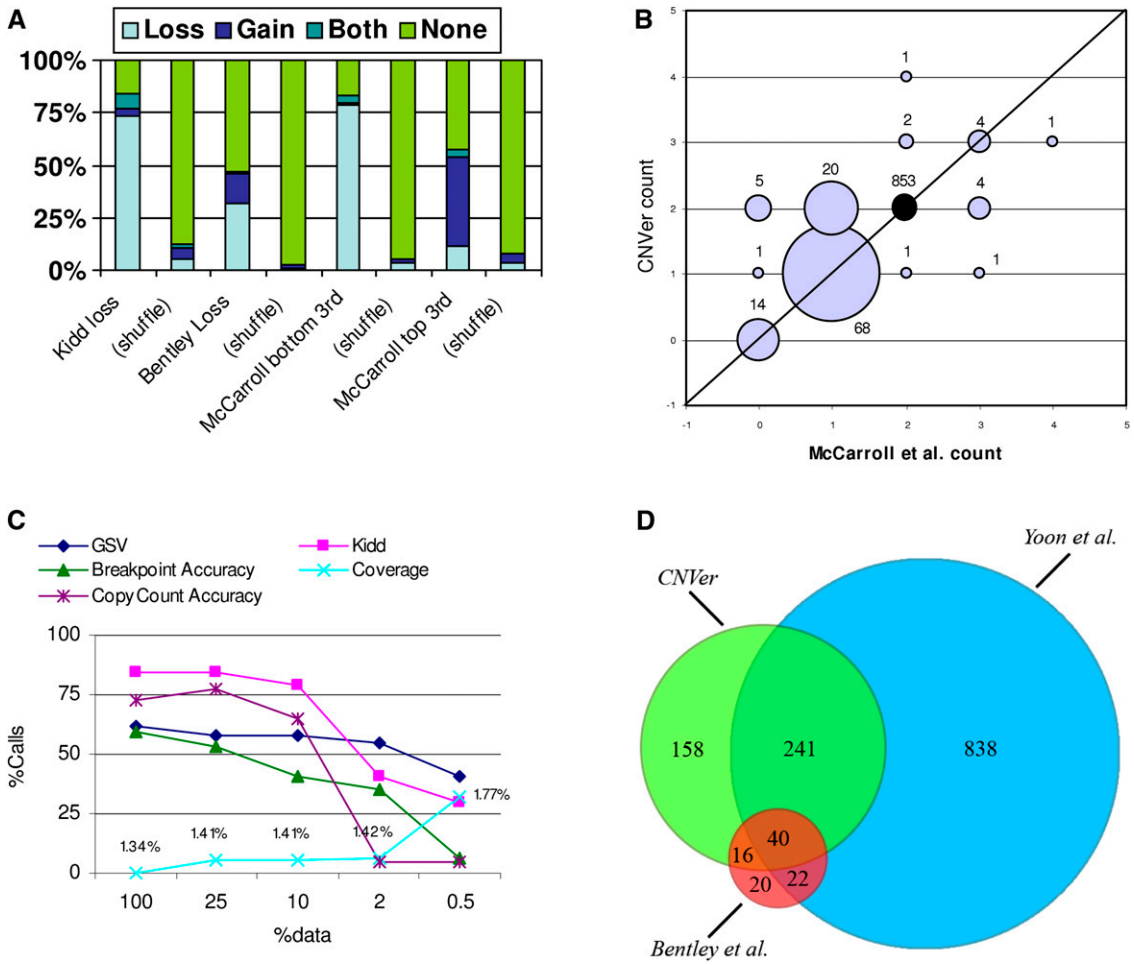


**Figure 2.** (A) Sensitivity analysis. We compared against four datasets: losses from Kidd et al. (2008) (141 calls) and Bentley et al. (2008) (1933 calls), as well as top and bottom 3rd quantiles from McCarroll et al. (2008) (93 and 26 calls, respectively). The chart shows the percentage of given calls that overlap only CNVer loss calls, only gain calls, both gain and loss calls, and no calls. In the case of Kidd et al., we require that the CNVer call is completely contained within the Kidd et al. call. We also make the same comparison against a randomly shuffled version of our calls. The raw numbers for this chart are included in Supplemental Table 5. (*B*) Count accuracy. A bubble chart comparing the copy counts reported by McCarroll et al. (2008) with those of CNVer for the 976 regions that do not overlap a segmental duplication. The area of each bubble is proportional to the number of regions with the given joint copy counts (except for the bubble with both counts being 2). One outlier is also not shown, where the CNVer copy count is 10 and the McCarroll et al. count is 2. The diagonal line represents regions where the predictions of the two methods matched. (*C*) Effect of number of reads. We measure the accuracy of our algorithm on datasets with 100%, 25%, 10%, 2%, and 0.5% of the original mate pairs. We show the percentage of called bases overlapping the GSV, the percentage of Kidd et al.'s calls that we overlap, the percentage of Kidd et al.'s sequenced variants that we detect with an F-score $\geq$ 0.9 (breakpoint accuracy), the percentage of McCarroll et al.'s regions (out of the 118 with copy count different from 2) for which CNVer's copy count agrees with McCarroll et al. (copy count accuracy), and the percent increase in the number of bases called with respect to the 100% run (coverage). The last series is also marked with the percent of the autosomal genome annotated as copy number variant. (*D*) Comparison against other methods. A three-way comparison of the calls made by CNVer, Yoon et al. (2009), and Bentley et al. (2008). Two calls are considered to overlap if they share at least one base pair. Note that the overlap measure is not symmetric, e.g., 14 of Yoon et al.'s calls overlap Bentley et al.'s and not CNVer, but 30 of Bentley et al.'s calls overlap Yoon et al.'s and not CNVer. What we show in the intersections, therefore, are the averages.

gains relative to the pool may be no-calls or losses relative to our reference (the case of losses is similar). Nevertheless, because of the 270 individuals, 90 were Yoruban and the remaining 180 were Eurasian, it is possible to use a population genomic argument to identify calls that are likely to be gains and losses for our individual. If the absolute copy count in NA18507 is strictly smaller (larger) than two-thirds of all samples, the reference genome (which mainly consists of DNA from individuals with a European ancestry) is likely to have a larger (smaller) copy count, and should correspond to our loss (gain) call. We identified those regions for which McCarroll et al. (2008) predicted a copy count for NA18507 that was strictly larger than two-thirds of the copy counts for the other samples (26 calls). Of these, we found that 58% overlapped one of our calls, with 80% overlapping a gain. Similarly, we identified regions with a copy count that was smaller than two-thirds of all samples (93 calls), and 83% overlapped one of our calls, with 99% of these overlapping a loss call. The overlap of all of these calls with the randomly shuffled version of our results was much lower (see Fig. 2A).

To check the specificity of our method, we used GSV, a database containing 8599 CNV regions found in 40 individuals using aCGH and covering 3.65% of the genome (http://projects.tcag.ca/variation/ng42m_cnv.php). We found that 57% of our calls overlap a call from GSV, compared with 6% of the randomly shuffled calls. Of the bases that we annotated as CNV, 62% are also in the

GSV. We also compare against the Database of Genomic Variants (DGV), which contains the results of many aCGH and PEM studies, totaling 19,792 calls and covering 25% of the genome. The variants predicted by CNVer showed strong correlation with this database, with 77% of our calls and 87% of our called bases overlapping a DGV call (see Supplemental Fig. 2).

### Count accuracy

In addition to predicting loss and gain regions, our method, for any given segment of the reference, can predict its absolute copy count in the donor (see Methods). We generated the counts for the variant regions identified by McCarroll et al. (2008) (as previously described) and compared them with McCarroll et al.'s copy count predictions (Fig. 2B; Supplemental Table 2). Because McCarroll et al.'s absolute copy counts are estimates from the relative intensities, it is less reliable in areas where most individuals have elevated copy counts (Alkan et al. 2009); therefore, we focused our analysis on the 976 regions that do not overlap segmental duplications. Our algorithm returned an identical copy count for 939 regions (96%). However, a significant fraction of these were baseline calls, where McCarroll and colleagues predicted a copy count of two. After removing these regions, 86 of the 118 remaining regions (73%) still had matching copy counts, while an additional 26 (22%) had a difference of one. We also saw greater concordance for lower copy count (McCarroll et al.'s copy count < 2) regions (82/108 = 76%) than for high copy count (>2) regions (4/10), indicating that higher copy counts might be more challenging to reconstruct than lower ones. Furthermore, we manually inspected the eight regions where the difference was more than one. In four of the cases, there is independent support for CNVer's copy counts from SNPs and known structural variants, while two other regions had evidence that both CNVer and McCarroll et al. (2008) made an off-by-one error. The final two regions did not have external supporting evidence for either McCarroll et al.'s or CNVer's copy counts (see Supplemental Text for a detailed analysis).

Additionally, we used the data of Alkan et al. (2009), who measured the absolute copy counts of several regions in NA18507 using fluorescence in situ hybridization (FISH) to further validate our algorithm. We generated copy counts for eight regions with reliable FISH measurements, and six out of eight matched the FISH results exactly, one region with a FISH copy count of 12 was predicted as 13, and one with a count of 16 as 20 (Supplemental Table 3).

### Breakpoint resolution

Kidd et al. (2008) fully sequenced 19 fosmid clones from NA18507 that they found to contain indels. These contain the actual donor sequence and give us a way to verify the breakpoint resolution of our method. Two of these are outside the detection scope of our method, containing novel insertions. The remaining 17 clones contained deletions varying in range from 1 kb to 229 kb (median 10 kb). We measure the extent to which our loss calls and the validated deletion region overlap using the F-score (Lewis and Gale 1994), where a score close to 0 indicates low overlap and a score of 1 indicates perfect overlap. The F-score is calculated as $F = 2(PR)/(P + R)$, where $P$ is the precision (percent of our call that overlaps the deleted area) and $R$ is the recall (percent of the deleted area which overlaps our call). Eight of the deletions have a score ≥0.99, representing near-perfect resolution. Two more have a strong correlation, with a score ≥0.90. Five more of the calls are over- or undercalled, and the final two are missed (no overlap; see Supplemental Table 4).

### Accuracy with less data

The DOC signature and the accuracy of linking clusters are directly related to the number of reads in the data set. Since the 3.6 G reads in our data set are expensive to obtain, we tested the robustness of our method by generating subsampled datasets, each containing a certain percentage of the original mate pairs, and rerunning CNVer (Fig. 2C; Supplemental Table 6). While overall the performance declines with the reduction in mate pairs, the specificity (fraction of the nucleotides in our calls that overlap GSV variants) remains relatively high, even with only 2% of the data. The sensitivity (percentage of Kidd et al. 2008 calls that we overlap) stays high at 10% of the data but drops-off significantly after that, despite an increase in the number of bases called by CNVer. Our ability to accurately determine breakpoints drops off steadily, since missing linking clusters make it difficult to precisely identify the borders of variants. The copy count concordance with McCarroll et al.(2008) stays high, even with 10% of the data (65%), but drops significantly at 2% of the data (5%) (Supplemental Table 6; Supplemental Fig. 3). Overall, these numbers indicate that CNVer's performance remains good with only 10% of the data, corresponding to an average of one read sampled every nine bases.

### Comparison with previous methods

There have been two genome-wide HTS-based studies performed on NA18507, both with the same data set utilized in the study. Yoon et al. (2009) use a DOC-based algorithm to identify CNVs on chromosome 1, and Bentley et al. (2008) use a PEM-based approach to identify deletions, as discussed previously. Table 1 and Figure 2D directly compare the predictions generated by the three methods for chromosome 1.

CNVer proved to be as sensitive as Yoon et al. (2009), with 82% (9/11) of Kidd et al. (2008) deletions recovered, while the total number of predictions was nearly three times smaller (435 vs. 1151). The lower specificity of Yoon et al. is also confirmed by comparing both datasets to GSV, with 73% of our calls (62% of our called bases) overlapping a GSV variant, and only 34% of Yoon et al.'s calls (54% of called bases) overlapping. Furthermore, the discrepancy between the call and base overlap percentages of Yoon et al. (34% vs. 54%) indicates that a large fraction of those calls that do not overlap GSV are short, and illustrates the difficulty of locating CNVs using the noisy DOC signal alone. The Bentley et al.

**Table 1. Comparison against other methods**

| | Calls | Coverage | Against GSV(by calls) | Against GSV(by bases) | Against Kidd et al. 2008 |
|---|---|---|---|---|---|
| CNVer | 435 | 1.75 | 73 | 62 | 82 |
| Yoon et al. 2009 | 1151 | 2.45 | 34 | 54 | 82 |
| Bentley et al. 2008 | 106 | 0.42 | 67 | 47 | 64 |

An analysis of the calls produced by CNVer, Yoon et al. (2009), and Bentley et al. (2008) for chromosome 1. We show the number of calls made, the percentage of chromosome 1 bases that are in a call (coverage), the percentage of calls that overlap GSV, the percentage of called bases that overlap GSV, and the percentage of Kidd et al. (2008) validated deletions that contain a call.

(2008) method is more specific and has the fewest calls of the three methods (fourfold less than CNVer, 10-fold less than Yoon et al. 2009). However, of the 11 validated Kidd et al. deletions on chromosome 1, Bentley et al. only recover seven (64%).

Figure 2D summarizes the overlap of the predictions made via the three different approaches. There are a large number of calls made by each of the methods that are unique, likely due to the complementarity of the approaches, though some fraction is also due to false-positives. Another possible reason is that for CNVs that occur in non-unique regions of the genome, it is impossible to assign the location of the CNV to one of the regions. For the purposes of validation, CNVer picks one of the regions arbitrarily, and similar behavior by the other methods may lead to calls that correctly identify the same CNV without overlapping on the reference.

## Discussion

In this study, we present CNVer: a method for predicting copy numbers and variants in the human genome from HTS data. Our approach combines paired-end mapping (PEM) and DOC analysis, and hence, is able to overcome some of the disadvantages of each of these methods. In particular, the insert size does not limit the size of the variants we can detect, as it typically does for PEM methods, which can only detect insertions up to the size of the mate pair insert (Tuzun et al. 2005; Bentley et al. 2008; Kidd et al. 2008). For HTS datasets, where the insert size is often <1 kb, this is critical, and, in fact, the recent PEM analysis of the same data set by Bentley et al. (2008) was not able to detect any insertions >200 b long. Another advantage of our method is that it can detect variants that do not create any discordant mate pairs. For example, if two or more tandem copies of a segment are present in the reference, having additional tandem copies in the donor does not lead to discordant mate pairs. Our method, however, can detect these variants based on the DOC of this region and the structure of the donor graph.

Additionally, since our algorithm requires a combination of support from DOC and PEM for making a call, it is more robust in the presence of false-positive indicators from either of the two approaches alone. Thus, we do not rely on having "uniquely best" read mappings, a limitation that has made it difficult for previous PEM techniques to detect variation within regions of segmental duplications (Tuzun et al. 2005; Korbel et al. 2007; Bentley et al. 2008; Kidd et al. 2008). Most variation occurs precisely within these regions (Locke et al. 2003; Iafrate et al. 2004; Sebat et al. 2004; Tuzun et al. 2005; Redon et al. 2006), which we confirm in our study. With short reads, the problem of multiple good mappings is further exacerbated. Our approach is able to mitigate this through the use of all good mappings for every mate pair, since spurious discordant mappings without support from DOC can be ignored by our algorithm.

Similarly, our approach is more robust in the presence of noise in the DOC signal. Campbell et al. (2008) first observed that regions exhibiting a sudden change in coverage depth correspond to the breakpoints indicated by discordant mate pairs. We take this idea a step further and use these mate pair breakpoints to delineate the windows used for calculating coverage depth (the edges of our donor graph). Our windows are not based on the DOC signal, allowing us to better mitigate the sequencing biases that cause uneven local coverage (Dohm et al. 2008; Harismendy et al. 2009). Thus, compared with previous DOC-based methods (Yoon et al. 2009), CNVer allows for a higher specificity without a reduction in sensitivity.

As any computational method, CNVer may mistakenly make false-positive calls in nonvariant regions. Our manual analysis has indicated that the most common cause of CNVer false-positive calls are missed linking clusters, leading to copy count predictions that disagreed with the DOC. Thus, one way to judge the confidence of a call is to look at the ratio between the number of normalized reads mapping to the region and the expected number. Gain calls where the ratio is less than one and loss calls where the ratio is more than one (together 4% of our calls) represent CNVer calls that are not supported by the DOC signal, and are possible false-positives. While the high concordance between CNVer predictions and structural variant databases (GSV and DGV) is strong evidence of the overall specificity of our results, wetlab validation is still necessary in order to ascertain the specific false-positives of CNVer and other methods.

Our method has several intrinsic limitations. Most importantly, though it can predict regions that have been gained in the donor, it cannot identify the locations where the duplicated sequence is inserted. While in some cases this limitation can be addressed through a manual analysis of the donor graph (e.g., Fig. 1B), often there are multiple plausible linking clusters that can be used to explain the change in the DOC. Another important limitation of our method is its inability to identify novel insertions—segments of the donor genome that do not match the reference. Several approaches have been proposed for this problem, including de novo (Birol et al. 2009) and reference-assisted (Chen et al. 2009) assemblies of the novel sequence. It may be possible to integrate one of these approaches with our data structure, using assembled contigs as edges within the donor graph. Building a full model that can correctly link inserted segments (whether duplicate or novel) with the location of the insertion is an important area for future work.

## Methods

Our algorithm consists of six stages: mapping reads, finding linking clusters, partitioning the reference genome, building the donor graph, defining and solving an optimization (flow) problem, and finally calling CNVs. Due to the large size of the donor graph, running the algorithm on the whole genome at once becomes computationally prohibitive, and therefore, except for the mapping stage, the algorithm is run separately for each chromosome. We describe each of the stages below.

### Stage 1: Mapping

We mapped the reads to the reference genome using Bowtie version 0.10.1 (Langmead et al. 2009) with parameters *"-v 2 -a -m 600–best–strata"*. This allows up to two mismatches, but only includes mappings with the minimum number of mismatches (e.g., if there is at least one exact mapping then only exact mappings will be included); also, any reads that map to more than 600 locations are omitted from the results (we refer to this set of read mappings as $M$). When working with mated reads, it is useful to consider a mate pair mapping, which is a pair of mappings, one for the forward and one, and one for the reverse read of the mate pair. A mate pair can have zero, one, or many mappings. We identify discordant mate pairs—those that do not have a concordant mapping. A concordant mapping is defined as a mate pair mapping with correct orientation, correct order, and distance that is less than $\mu + 3\sigma$ ($\mu$ = 208 and $\sigma$ = 13 are the mean and standard deviation of the insert size in our data set). We let $D$ be the subset of mate pair mappings that belong to discordant mate pairs. Thus, our algorithm works

with two sets of mappings: a set of read mappings $M$, which is later used for DOC calculation, and a set of mate pair mappings $D$, which is later used to identify linking clusters.

### Stage 2: Finding linking clusters

Discordant mate pairs that are consistent with a single variant will map with a similar mapped distance, order, and orientation to the reference. We identify the order and orientation of a mate pair mapping using a shorthand notation. For example, a mapping that is [+−] has the left read mapping to the positive strand and the right read mapping to the negative strand. In this way, we characterize every mapping from $D$ as being of type [+−], type [−+], type [++], or type [−−]. To find the linking clusters, we partition $D$ into four subsets according to their types, and cluster each subset independently. Ideally, each cluster collects the mappings of all of the reads that span a single donor breakpoint, allowing for the fact that the insert size can deviate $\pm$ 3$\sigma$ (mate pairs with larger deviations are ignored). To begin, the algorithm notes the locations (*left, right*) of the leftmost mate pair mapping. It then makes a cluster that includes all the mappings *(left′, right′)* such that $left \leq left' \leq left + \mu + 3\sigma$ and $|left' - left - (right - right')| \leq 6\sigma$ for type [++] and [−−] clusters and $|left' - left - (right' - right)| \leq 6\sigma$ for type [+−] and [−+] clusters. Each of the mappings in this cluster is marked as used and never again considered by the algorithm. The algorithm then picks the leftmost mate pair mapping that has not yet been used and repeats the process. This continues until all of the mappings have been used. Note that a single mate pair may be involved in multiple clusters via its different mappings; however, a single mate pair mapping cannot be in more than one cluster. We discard all clusters with a distance between the left and right mates greater than 10 Mb. Finally, all clusters that do not contain enough mappings are discarded (13 mappings for the full data set, three for the 25% subsample, and two for the remaining subsamples). In total, there were 646,000 type [+−], 641 type [−+], 298 type [++], and 302 type [−−] clusters for the full data set.

Next, for each cluster we identify its innermost positions—the locations that would have been closest to the donor's breakpoint. Though we do not explicitly know the location of the reads in the donor, we can infer the relative order from the type of cluster. For [+−] and [−+] clusters, this is the right end of the rightmost positive read mapping and the left end of the leftmost negative read mapping. For [++] clusters, it is the right end of the rightmost read mapping, and the right end of the rightmost read mapping taken out of the left reads of the mate pairs (see Fig. 3 for a pictorial depiction, and for the definition for type [−−] clusters).

This clustering algorithm is efficiently implemented in *O(D log D)* time using a sort followed by a simple left-to-right scan of $D$. The output of the algorithm is the set of variables *linClu(i,s)* and *linClu(i,t)*, which represent the location of the two innermost positions for cluster $i$ (which of the positions is $s$, and which is $t$ is shown in Fig. 3), and the variables *linCluType(i)*, which represent the type of the $i$th cluster.

### Stage 3: Partitioning the reference

For duplications in regions that already have multiple copies in the reference genome, we usually cannot identify which of the copies has been duplicated. For reads sampled from one of these regions, it is likewise usually impossible to determine their origin. Therefore, we group together highly similar regions of the reference into blocks, and allow our algorithm to work with these blocks rather than with the individual regions.

To identify these blocks we start with all maximal pairwise local alignments of the reference to itself (called the self-alignment),

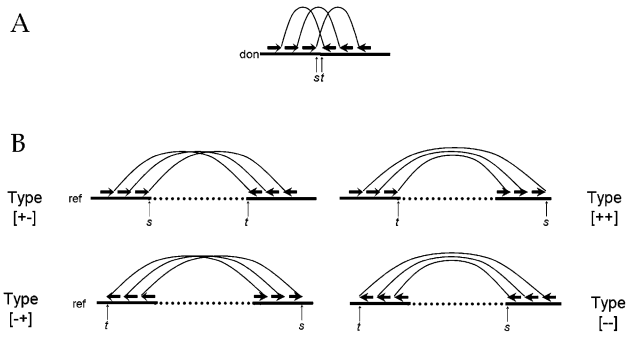


**Figure 3.** Clustering. (*A*) A cluster originates in the donor, where the mate pairs of the cluster are all of those that span a given location. The *innermost* positions of this cluster, specified by $s$ and $t$, are (nearly) adjacent. If this adjacency doesn't exist in the reference, then the cluster will map discordantly. Nevertheless, all of the mate pairs within the cluster will have a similar mapped distance, order, and orientation. There are four distinct types of clusters, based on the order and orientation of the mate pair mappings, as shown in *B*. In each case, we can identify the locations $s$ and $t$ in the reference, and mark them as a putative adjacency in the donor.

where each alignment must contain at least 100 nonrepeat masked bases and have ≥99% similarity (downloaded from http://hgdownload.cse.ucsc.edu/goldenPath/hg18/vsSelf/). Next, we take any alignment that spans an endpoint of another alignment and split it at that point. This is repeated until convergence, when there are no more alignments that can be split. In this way, no two alignments will involve partially overlapping regions. Next, we transitively expand the set of alignments. That is, if there are alignments between A and B and between B and C, then we add an alignment between A and C, if one does not already exist. This is repeated until the set of all alignments is transitive.

The endpoints in the final set of alignments induce a partition of the reference into regions. Furthermore, because the alignments are transitive, these regions are organized into equivalence classes, according to the alignments. Each equivalence class, which we call a block, represents a set of highly similar regions.

The output of this stage of the method is stored using variables as follows: The $j$th region of the $i$th block is stored as the closed interval *partition(i,j)*. Furthermore, for each block, we label the endpoints of each of its regions with $s$ and $t$ in such a way that the $s$ ends of all the regions represent the same endpoint of the multiple alignment. We thus have the variables *partition(i,j,s)* and *partition(i,j,t)* to represent the endpoints of each region.

### Stage 4: Building the donor graph

Given the partition of the genome and the linking clusters, our algorithm next builds the donor graph, which is bidirected. In bidirected graphs, each edge has two separate orientations, one associated with each vertex, as opposed to directed graphs where an edge has only one orientation. Allowing different orientations captures the fact that DNA is double stranded, and that we do not know which strand each read comes from (for more background, see Kececioglu 1991; Medvedev and Brudno 2009). The idea behind the donor graph is that it represents the donor genome using the available information—namely, the adjacency information about the blocks, both via the reference sequence and the linking clusters. The endpoints of each block $i$ are represented with vertices called $v_r(i,s)$ and $v_r(i,t)$, and the block itself with a sequence edge between them. This edge is potentially broken up into a path by "entry/exit" points for donor edges (we call these the $v_d$ vertices).

These, in effect, subdivide each block into sub-blocks. Furthermore, any two $v_r$ vertices are connected if they represent adjacent locations in the reference, and any two $v_d$ vertices are connected if they correspond to the same linking cluster. Figure 4 shows a toy example of a donor graph, while for our data set, the donor graph of chromosome 1 contains 175 thousand vertices and 351 thousand edges. We now give a formal definition.

The vertex set is the union of partition vertices $\{v_r(i, end)$ such that $1 \le i \le n_{blk}, end \in \{s,t\}\}$ and linking vertices $\{v_d(i, end)$ such that $1 \le i \le n_{clu}, end \in \{s,t\}\}$, where $n_{blk}$ and $n_{clu}$ are the number of blocks and of linking clusters. Thus, for every block and every linking cluster, we have an $s$ and a $t$ vertex, and every vertex represents a set of genomic locations. Next, for each block $i$, let *brk(i)* be the following set of pairs sorted in increasing order of the second element:

$$\{(v_d(k, end), |linClu(k, end) - partition(i, j, s)|)$$
$$\text{such that } linClu(k, end) \in partition(i, j)\}$$

This sequence contains all of the linking cluster endpoints that are contained in some interval $j$ of the $i$th block, together with their offset from the $s$ end of the interval. We refer to the first element of the $x$th pair of *brk(i)* as *brk(i,x)*. We now build the edgeset as follows.

1. Within every block $i$, we chain together all the vertices whose location is in $i$. Formally, if *brk(i)* is empty, we make an edge out of $v_r(i,s)$ and into $v_r(i,t)$. Otherwise, for every $1 \le x \le |brk(i)| - 1$, we make an edge out of *brk(i,x)* and into *brk(i,x + 1)*. We also add an edge out of $v_r(i,s)$ and into *brk(i,1)* and an edge out of *brk(i, |brk(i)|)* and into $v_r(i,t)$. Intuitively, each edge corresponds to a portion of the $i$th block, and thus to a set of similar sequences of the reference. These edges are therefore referred to as sequence edges.
2. Next, we connect any partition vertices which contain adjacent locations. Formally, for each *end, end′, i, i′, j, j′*, if *partition(i′, j′,end′) − partition(i,j,end) = 1*, then we make an edge between $v_r(i,end)$ and $v_r(i′,end′)$. The directionality of the edge is given solely by *end* and *end′*, as follows. If *end* = $s$ (respectively, $t$) then the edge goes into (respectively, out of) $v_r(i,end)$, and if *end′* = $s$ (respectively, $t$) then the edge goes into (respectively, out of) $v_r(i′,end′)$. We refer to these edges as reference edges, because they represent adjacencies present in the reference.
3. Finally, we connect the linking vertices associated with each cluster. Formally, for each linking cluster $i$, we add an edge between $v_d(i,s)$ and $v_d(i,t)$. The directionality of the edge is given by the type of the cluster. Types [+−] and [−+] correspond to the edge going out of $s$ and into $t$, type [++] to going out of both $s$ and $t$, and type [−−] going into both $s$ and $t$. We refer to these edges as donor edges, because they represent adjacencies that are putatively present in the donor.

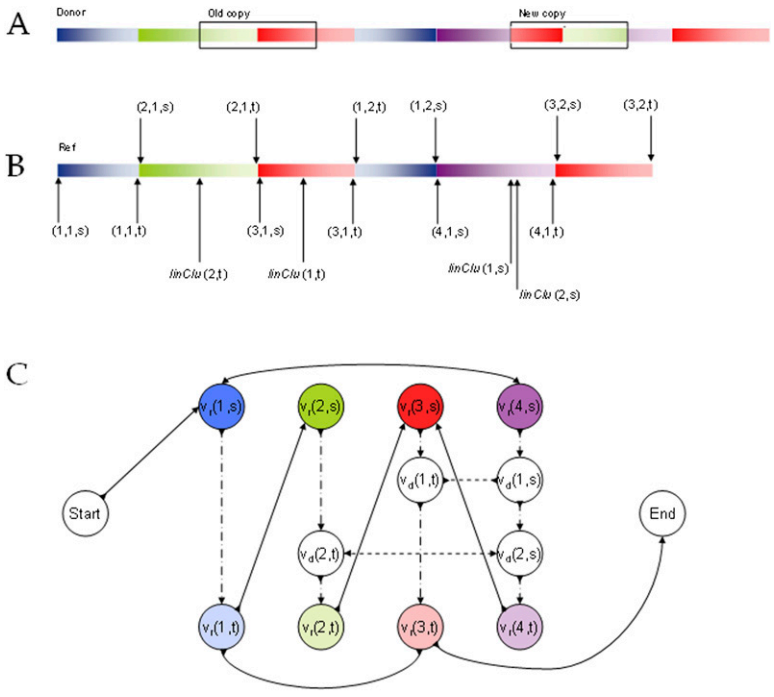


**Figure 4.** Donor graph. We show a toy example of a donor genome (*A*) and a reference (*B*). Identical regions have the same color, with inversions having a reverse color gradient. The donor differs from the reference only in that there is one nontandem inverted duplication, as shown in *A*. (*B*) The partitioning of the reference, indicated by triplets; for example, (3,1,*s*) refers to the location of the *s* endpoint of the first region of the third block. There are four blocks in the partition: blue (1), green (2), red (3), and violet (4). There are two regions in the blue block, each with different directionality; and there are two regions in the red block, with the same directionality. There are also two linking clusters, and their *s* and *t* endpoints are indicated by *linClu*. It is not illustrated, but cluster 1 is of type [++] and cluster 2 is of type [−−]. It is easy to check that the two linking clusters correspond with the duplication in the donor, both in their location and their type. (*C*) The donor graph. There are two special Start and End vertices, signifying the start and end of the genome. The other uncolored vertices are the donor vertices; the colored vertices are the partition vertices, with the darker color representing the *s* endpoints and the lighter the *t* endpoints. Sequence edges are shown with long dashed lines, donor edges with short dashed lines, and reference edges with regular lines.

Finally, we add two special Start and End vertices to the graph, symbolizing the beginning and end of the genome, an edge out of Start into the partition vertex with the leftmost location, and an edge out of the partition vertex with the rightmost location and into End. By walking along this graph from Start to End and concatenating the sequences associated with each sequence edge, we can spell a genomic sequence. For example, the reference itself is spelled by some walk in this graph, since every two regions adjacent in the reference are connected by a reference edge. The key desired property of the graph, however, is that each of the donor haplotypes, which contain both the reference and donor adjacencies, can be spelled by a walk (with the exception of any novel, donor-specific sequence).

### Stage 5: Finding flow

Having constructed the donor graph, one possible goal is to find which of the many possible walks corresponds to the donor genome. However, this amounts to assembling the whole donor, which is a much more difficult task than discovering CNVs. For our purposes, we need to find only the copy counts of each sub-block, which are given by the traversal counts of a walk on the sequence edges. We therefore model this as a minimum cost flow problem, a much easier optimization task.

A flow is a function $f$ that assigns a non-negative integer $f_e$ to each edge of the graph such that for each vertex except Start and End, the flow along the in-edges is equal to the flow along the out-edges

(the balance property). For Start (End), the sum of the flows along the out-edges (in-edges) is fixed to some constant (this value is referred to as the size of the flow). Any given walk $w$ from Start to End induces a traversal count on the edges that satisfy the balance property, since a walk must enter any vertex the same number of times it exits it. Therefore, the traversal counts of $w$ correspond to a flow $f$. On the other hand, any flow $f$ corresponds to many walks whose traversal counts are given by $f$. All of these walks, however, will have the same genomic copy counts, and so, for the purposes of CNV prediction they are identical. Our optimization task is to find a flow of size 2 (corresponding to the two haplotypes) in the donor graph with maximum score.

### *Probabilistically scoring flows*

We define a score function on any flow through the donor graph. The ratio between the number of reads mapping to each sub-block and its length suggests the number of times that segment appears in the donor. Informally, our score measures the faithfulness of the flow copy counts to those suggested by the DOC.

We begin by identifying regions that have an extremely high copy count in the reference; for these regions, we do not use the DOC signal because we expect too much noise. We simulate an error-free read at every position and map it back to the chromosome, allowing up to two mismatches. For every read that has more than 400 mappings, we mark all of the hits' starting positions as high-copy reference regions.

For every position, the DOC is defined as the number of reads with a mapping in $M$ that starts at that position, where the contribution from each read is normalized by the number of mappings it has in $M$. The expected DOC is the number of reads divided by the length of the genome; however, because of sequencing bias, we calculate the expected DOC separately for regions with different GC-content (using 10 bins). Notice that because we normalize each read in computing the DOC, it should be close to the expected in nonvariant regions, including nonunique ones. We also identify spikes, which are nonrepeat masked locations with DOC over 15 times the expected. Though such regions may occur when, for example, the donor has 60 copies and the reference only four, it is much more likely that they have been oversampled due to sequencing bias.

The DOC for a genomic region $r$ is the sum of the DOC for each point location in $r$ that is not in a contig break, high-copy reference region, or a spike. The effective length of $r$ is its length minus the length of the ignored regions. For a sequence edge $e$, the effective length $l_e$ is the average of the effective lengths of each region associated with $e$, while the DOC $k_e$ is the sum of the DOCs in these regions. By modeling the sequencing as a Poisson process, we can describe the likelihood that the observed DOC along $e$ is $k_e$ given that it appears $f_e$ times in the donor by

$$\frac{e^{-\lambda_e f_e l_e}(\lambda_e f_e l_e)^{k_e}}{k_e!},$$

where $\lambda_e$ is the expected DOC for a position with $e$'s GC-content. The score function is the product of the likelihoods that each of the sequence edges (sub-blocks) appears $f_e$ times in the donor, which is

$$L(f) = \prod_e \frac{e^{-\lambda f_e l_e}(\lambda f_e l_e)^{k_e}}{k_e!}.$$

### *Finding the highest scoring flow*

Given the donor graph, the observed DOC $k_e$, the expected DOC $\lambda_e$, and the effective length $l_e$ for every sequence edge, the task is to find a flow of size 2 that maximizes $L(f)$. Finding a flow that maximizes $L(f)$ is the same as finding a flow that minimizes

$$\sum \lambda_e f_e l_e - k_e \ln(\lambda_e f_e l_e),$$

which is the negative log of $L(f)$ with terms independent of $f_e$ removed. This cost function is a linear combination of convex functions for each sequence edge, and the optimization problem can therefore be expressed as a min-cost bidirected flow problem (Ahuja et al. 1993; Lawler 2001). Though exact polynomial-time algorithms exist (Gabow 1983), they do not have efficient implementations; we therefore use the monotonization algorithm that we have previously found to work very well in practice (Medvedev and Brudno 2009). Briefly, the algorithm works by reducing the min-cost flow problem in bidirected graphs to one in regular directed graphs, for which there exist efficient solvers; we use the CS2 package of Goldberg (1997). The flow in the directed graph is then used to assign the flow to the original bidirected graph.

### Stage 6: Calling variants and predicting copy counts

After solving the flow problem, we predict CNVs by finding walks in the graph where the flow is consistently higher or lower than the corresponding copy counts in the reference (number of regions in a sub-block). The walks are found greedily by reporting the longest walk first, adjusting by one the flow along it, and then repeating the process. For each walk, we report the genomic region that the walk represents. If there is more than one such region (the case of an area that appears multiple times in the reference), we assign the variant to one of them arbitrarily. We do not report calls that have less than 1000 bases in non-high-copy reference regions. Also, we join together calls of the same type (gain/loss) when the gap between them is call-free and <5% of their joined length (repeated iteratively until convergence).

In addition to predicting CNV regions, our method, for any given segment of the reference, can predict its absolute copy count in the donor. Given such a segment, CNVer identifies the corresponding sequence edges, and reports the length-weighted median of the flow along them.

## Acknowledgments

We thank Seunghak Lee and Adrian Dalca for their help during this project, and Can Alkan and Fereydoun Hormozdiari for providing mappings that were used in the earlier development of the algorithm. We also thank the anonymous reviewers for helping to significantly improve the quality of this work.

## References

Ahuja RK, Magnanti TL, Orlin JB. 1993. *Network flows: Theory, algorithms, and applications*. Prentice-Hall, Upper Saddle River, NJ.

Alkan C, Kidd J, Marques-Bonet T, Aksay G, Antonacci F, Hormozdiari F, Kitzman JO, Baker C, Malig M, Mutlu O, et al. 2009. Personalized copy number and segmental duplication maps using next-generation sequencing. *Nat Genet* **41:** 1061–1067.

Bentley DR, Balasubramanian S, Swerdlow HP, Smith GP, Milton J, Brown CG, Hall KP, Evers DJ, Barnes CL, Bignell HR, et al. 2008. Accurate whole human genome sequencing using reversible terminator chemistry. *Nature* **456:** 53–59.

Birol I, Jackman SD, Nielsen CB, Qian JQ, Varhol R, Stazyk G, Morin RD, Zhao Y, Hirst M, Schein JE, et al. 2009. De novo transcriptome assembly with ABySS. *Bioinformatics* **25:** 2872–2877.

Campbell P, Stephens P, Pleasance E, O'Meara S, Li H, Santarius T, Stebbings L, Leroy C, Edkins S, Hardy C, et al. 2008. Identification of somatically acquired rearrangements in cancer using genome-wide massively parallel paired-end sequencing. *Nat Genet* **40:** 722–729.

Carter N. 2007. Methods and strategies for analyzing copy number variation using DNA microarrays. *Nat Genet* **39:** S16–S21.

Chen K, Wallis JW, McLellan MD, Larson DE, Kalicki JM, Pohl CS, McGrath SD, Wendl MC, Zhang Q, Locke DP, et al. 2009. BreakDancer: An algorithm for high-resolution mapping of genomic structural variation. *Nat Methods* **6:** 677–681.

Chiang DY, Getz G, Jaffe DB, O'Kelly MJ, Zhao X, Carter SL, Russ C, Nusbaum C, Meyerson M, Lander ES. 2009. High-resolution mapping of copy-number alterations with massively parallel sequencing. *Nat Methods* **6:** 99–103.

Cooper GM, Zerr T, Kidd JM, Eichler EE, Nickerson DA. 2008. Systematic assessment of copy number variant detection via genome-wide SNP genotyping. *Nat Genet* **40:** 1199–1203.

Dohm JC, Lottaz C, Borodina T, Himmelbauer H. 2008. Substantial biases in ultra-short read data sets from high-throughput DNA sequencing. *Nucleic Acids Res* **36:** e105. doi: 10.1092/nat/gkn425.

Fiume M, Williams V, Brook A, Brudno M. 2010. Savant: Genome browser for high throughput sequencing data. *Bioinformatics* **26:** 1938–1944.

Gabow HN. 1983. An efficient reduction technique for degree-constrained subgraph and bidirected network flow problems. In *Proceedings of the 15th Annual ACM Symposium on Theory of Computing*, pp 448–456. ACM, New York.

Goldberg AV. 1997. An efficient implementation of a scaling minimum-cost flow algorithm. *J Algorithms* **22:** 1–29.

Harismendy O, Ng P, Strausberg R, Wang X, Stockwell T, Beeson K, Schork N, Murray S, Topol E, Levy S, et al. 2009. Evaluation of next generation sequencing platforms for population targeted sequencing studies. *Genome Biol* **10:** R32. doi: 10.1186/gb-2009-10-3-r32.

Hormozdiari F, Alkan C, Eichler EE, Sahinalp SC. 2009. Combinatorial algorithms for structural variation detection in high-throughput sequenced genomes. *Genome Res* **19:** 1270–1278.

Iafrate AJ, Feuk L, Rivera MN, Listewnik ML, Donahoe PK, Qi Y, Scherer SW, Lee C. 2004. Detection of large-scale variation in the human genome. *Nat Genet* **36:** 949–951.

Kececioglu J. 1991. *Exact and approximation algorithms for DNA sequence reconstruction*. Technical Report 91-26. University of Arizona, Tuscon.

Kidd JM, Cooper GM, Donahue WF, Hayden HS, Sampas N, Graves T, Hansen N, Teague B, Alkan C, Antonacci F, et al. 2008. Mapping and sequencing of structural variation from eight human genomes. *Nature* **453:** 56–64.

Korbel JO, Urban AE, Affourtit JP, Godwin B, Grubert F, Simons JF, Kim PM, Palejev D, Carriero NJ, Du L, et al. 2007. Paired-end mapping reveals extensive structural variation in the human genome. *Science* **318:** 420–426.

Langmead B, Trapnell C, Pop M, Salzberg SL. 2009. Ultrafast and memory-efficient alignment of short DNA sequences to the human genome. *Genome Biol* **10:** R25. doi: 10.1186/gb-2009-10-3-r25.

Lawler EL. 2001. *Combinatorial Optimization: Networks and matroids*. Dover Publications, Mineola, NY.

Lee S, Cheran E, Brudno M. 2008. A robust framework for detecting structural variations in a genome. *Bioinformatics* **24:** i59–i67.

Lee S, Hormozdiari F, Alkan C, Brudno M. 2009. MoDIL: Detecting small indels from clone-end sequencing with mixtures of distributions. *Nat Methods* **6:** 473–474.

Lewis DD, Gale WA. 1994. A sequential algorithm for training text classifiers. In *Proceedings of the 17th annual international ACM SIGIR conference on research and development in information retrieval*, pp 3–12. Springer-Verlag, New York.

Locke D, Archidiacono N, Misceo D, Cardone M, Deschamps S, Roe B, Rocchi M, Eichler E. 2003. Refinement of a chimpanzee pericentric inversion breakpoint to a segmental duplication cluster. *Genome Biol* **4:** R50. doi: 10.1186/gb-2003-4-8-r50.

McCarroll SA, Altshuler DM. 2007. Copy-number variation and association studies of human disease. *Nat Genet* **39:** S37–S42.

McCarroll SA, Kuruvilla FG, Korn JM, Cawley S, Nemesh J, Wysoker A, Shapero MH, de Bakker PI, Maller JB, Kirby A, et al. 2008. Integrated detection and population-genetic analysis of SNPs and copy number variation. *Nat Genet* **40:** 1166–1174.

McKernan KJ, Peckham HE, Costa GL, McLaughlin SF, Fu Y, Tsung EF, Clouser CR, Duncan C, Ichikawa JK, Lee CC, et al. 2009. Sequence and structural variation in a human genome uncovered by short-read, massively parallel ligation sequencing using two-base encoding. *Genome Res* **19:** 1527–1541.

Medvedev P, Brudno M. 2009. Maximum likelihood genome assembly. *J Comput Biol* **16:** 1–16.

Medvedev P, Stanciu M, Brudno M. 2009. Computational methods for discovering structural variation with next-generation sequencing. *Nat Methods* **6:** S13–S20.

Paten B, Herrero J, Beal K, Fitzgerald S, Birney E. 2008. Enredo and Pecan: Genome-wide mammalian consistency-based multiple alignment with paralogs. *Genome Res* **18:** 1814–1828.

Pevzner PA, Tang H, Tesler G. 2004. De novo repeat classification and fragment assembly. *Genome Res* **14:** 1786–1796.

Raphael BJ, Volik S, Collins C, Pevzner PA. 2003. Reconstructing tumor genome architectures. *Bioinformatics* **19:** 162–171.

Raphael B, Zhi D, Tang H, Pevzner P. 2004. A novel method for multiple alignment of sequences with repeated and shuffled elements. *Genome Res* **14:** 2336–2346.

Redon R, Ishikawa S, Fitch KR, Feuk L, Perry GH, Andrews TD, Fiegler H, Shapero MH, Carson AR, Chen W, et al. 2006. Global variation in copy number in the human genome. *Nature* **444:** 444–454.

Sebat J, Lakshmi B, Troge J, Alexander J, Young J, Lundin P, Maner S, Massa H, Walker M, Chi M, et al. 2004. Large-scale copy number polymorphism in the human genome. *Science* **305:** 525–528.

Simpson JT, McIntyre RE, Adams DJ, Durbin R. 2010. Copy number variant detection in inbred strains from short read sequence data. *Bioinformatics* **26:** 565–567.

Tuzun E, Sharp AJ, Bailey JA, Kaul R, Morrison VA, Pertz LM, Haugen E, Hayden H, Albertson D, Pinkel D, et al. 2005. Fine-scale structural variation of the human genome. *Nat Genet* **37:** 727–732.

Volik S, Zhao S, Chin K, Brebner JH, Herndon DR, Tao Q, Kowbel D, Huang G, Lapuk A, Kuo W, et al. 2003. End-sequence profiling: Sequence-based analysis of aberrant genomes. *Proc Natl Acad Sci* **100:** 7696–7701.

Yoon S, Xuan Z, Makarov V, Ye K, Sebat J. 2009. Sensitive and accurate detection of copy number variants using read depth of coverage. *Genome Res* **19:** 1586–1592.

*Received February 8, 2010; accepted in revised form August 24, 2010.*